AFRL-IF-RS-TR-2006-270
Final Technical Report
August 2006

SIMULATION DEVELOPMENT FOR DYNAMIC SITUATION AWARENESS AND PREDICTION II

Northrop Grumman Mission Systems

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

STINFO FINAL REPORT

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-2006-270 has been reviewed and is approved for publication.

APPROVED: /s/

DAWN A. TREVISANI
Project Engineer

FOR THE DIRECTOR: /s/

JAMES W. CUSACK
Chief, Information Systems Division
Information Directorate

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
AUG 2006	Final	Apr 05 – Jun 06

4. TITLE AND SUBTITLE

SIMULATION DEVELOPMENT FOR DYNAMIC SITUATION AWARENESS AND PREDICTION II

5a. CONTRACT NUMBER FA8750-05-C-0087

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER 62702F

6. AUTHOR(S)

Kevin C. Trott

5d. PROJECT NUMBER 459S

5e. TASK NUMBER CO

5f. WORK UNIT NUMBER AM

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Northrop Grumman Mission Systems
Beeches Technical Campus, Rt 26N
Rome NY 13440

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

AFRL/IFSB
525 Brooks Rd
Rome NY 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-IF-RS-TR-2006-270

12. DISTRIBUTION AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED. PA# 06- 585

13. SUPPLEMENTARY NOTES

14. ABSTRACT
The objective of this effort was to create a “closed-loop” simulation environment in which detailed mission plans can be developed, used as input to a set of distributed simulations, and be executed within the simulation environment. These simulations provide feedback to prototype C4ISR systems in the form of mission status reports, sensor tracks, and other ISR mission results reports, which can be used to maintain situation awareness and to dynamically adjust mission plans in response to events. This work was carried out within the context of the Joint Synthetic Battlespace for Research and Development (JSB-RD), a distributed C4ISR modeling and simulation environment at the AFRL Rome Research Site. The JSB-RD distributed simulation environment includes Joint Semi-Automated Forces (JSAF) simulation, a weather federate, a dynamic terrain federate, a culture/clutter simulation, a visualization architecture for controlling the simulation/mission and the GIESim communication simulation which provides Link-16 network modeling. The environment also includes tools for creating scenarios from existing Air Battle Plans, extracted from the Air Operations Data Base (AODB) within the Theater Battle Management Core System (TBMCS), as well as gateways for connecting simulations communicating using DMSO’s HLA and/or the earlier DIS protocol, with C4ISR systems.

15. SUBJECT TERMS
Model Abstraction, Simulation Abstraction, C4ISR, ISR, Modeling and Simulation

16. SECURITY CLASSIFICATION OF: a. REPORT	b. ABSTRACT	c. THIS PAGE	17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
U	U	U	UL	51	Dawn Trevisani

19b. TELEPONE NUMBER (*Include area code*)

TABLE OF CONTENTS

Section **Page**

LIST OF FIGURES

Figure **Page**

1. INTRODUCTION

This final technical report summarizes the accomplishments and lessons learned under the Simulation Development for Dynamic Situation Awareness and Prediction II (Sim Dev for DSAP II) contract. This work was performed by Northrop Grumman Mission Systems (NGMS) for the AFRL, C4ISR Modeling and Simulation Branch (IFSB), under Contract FA8750-C-05-0087. This report addresses CLIN 0002, CDRL A006 of that contract.

The objective of this effort is to create a “closed-loop” simulation environment, in which detailed mission plans can be developed, used as input to a set of distributed simulations, and be executed within the simulation environment. These simulations provide feedback to prototype Command, Control, Communications, Computers, Intelligence, Surveillance, and Reconnaissance (C4ISR) systems in the form of mission status reports, sensor tracks, and other ISR mission results reports, which can be used to maintain situation awareness and to dynamically adjust mission plans in response to events. This work was carried out within the context of the Joint Synthetic Battlespace for Research and Development (JSB-RD), a distributed C4ISR modeling and simulation environment at the AFRL Rome Research Site. The JSB-RD synthetic battlespace is designed to address three objectives:

1. To provide a testbed for C4ISR research, experimentation, and evaluation, supporting AFRL C4ISR concepts and programs such as Predictive Battlespace Awareness (PBA), the Commander’s Predictive Environment (CPE), Effects Based Operations (EBO), and Airborne Networking Technology (ANT).

2. To explore the synchronization of simulations to real-world situation data in order to predict future events, supporting concepts such as Dynamic Situation Awareness and Prediction (DSAP) and Course of Action (COA) Analysis.

3. To enable research into simulation science, particularly in the areas of adversarial modeling, multi-resolution modeling, and visualization.

The JSB-RD distributed simulation environment was constructed primarily by integrating existing simulations and tools. The JSB-RD environment is currently centered on the Joint Semi-Automated Forces (JSAF) simulation software. JSAF is a computer generated forces (CGF) system that is used by the U.S. Joint Forces Command for joint experimentation. The JSB-RD environment also includes the Ocean, Atmosphere, and Space Environmental Services (OASES) system, which models weather, and the Dynamic Terrain Simulation (DTSim), which models changes to the environment such as bomb craters, damage to buildings, and the creation and destruction of obstacles, as well as a culture/clutter simulation, which models civilian vehicle and personnel traffic. The environment also includes tools for creating scenarios from existing Air Battle Plans, extracted from the Air Operations Data Base (AODB) within the Theater Battle Management Core System (TBMCS), as well as gateways for connecting simulations communicating using DMSO’s High Level Architecture (HLA) and/or the earlier Distributed Interactive Simulation (DIS) protocol, with C4ISR systems, using a variety of different mechanisms.

Under this effort, the Global Information Enterprise Simulation (GIESim) communication simulation was added to the JSB-RD environment to provide Link-16 network modeling. GIESim determines whether or not entities modeled by JSAF can communicate with one another, and, if so, the communication latency.

Also under this effort, the JSB-RD environment was used to support the development of DARPA's Dynamic Network Centric Warfare (DNCW) concept. The JSB-RD environment was used to produce visualizations to help illustrate how the DNCW concept might be implemented in an urban environment.

Support for other AFRL/IFSB efforts: RAM Labs, Synergia, etc.

Section 2 describes the JSB-RD environment in more detail. Section 3 discusses the activities that were performed under this effort, including the JSAF-GIESim integration, DNCW visualization, and TBMCS Track Management Data Base stimulation efforts. Section 4 summarizes the lessons learned in the course of performing this effort, including recommendations for subsequent related work. Section 5 contains a list of acronyms.

2. JSB-RD DISTRIBUTED SIMULATION ENVIRONMENT

As noted above, the JSB-RD distributed simulation environment consists of a number of components, which are connected together using several different mechanisms. Fundamentally, the environment is an HLA federation made up of simulations that communicate using a variation of the Joint Urban Operations (JUO) Federation Object Model (FOM). Simulations that still use the DIS protocol can be connected through a DIS-HLA gateway federate. Similarly, an HLA-JBI gateway allows the federation to communicate with C4ISR system component prototypes being developed at AFRL. Finally, selected Link-16 messages can be generated using an Army software application named SIMPLE.

2.1 Overall Architecture

The overall architecture of the JSB-RD distributed simulation environment is shown in Figure 2-1. It consists of three primary components, addressing each of the three primary aspects of any simulation environment.

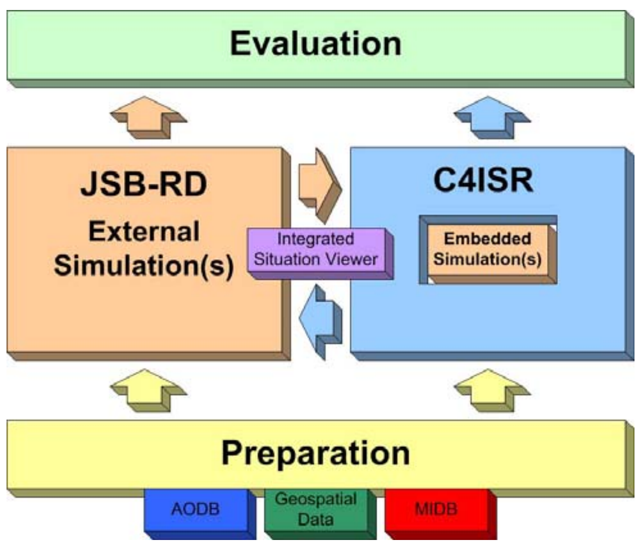


Figure 2-1: JSB-RD Overall Architecture

At the bottom, the Preparation component addresses experiment planning, scenario preparation, and all other aspects of preparing a simulation experiment. This includes accessing, extracting, and manipulating various kinds of source data, including Air Battle Plan (ABP) and Friendly Air Order of Battle information contained in the AODB, and Enemy Order of Battle (EOB) information contained in the MIDB, as well as geospatial data. This also includes emulating the detailed mission planning that normally is performed at the unit level. In the middle are the components that address the execution phase of a simulation experiment. These consist of the various simulations that make up the JSB-RD environment, as well as real C4ISR systems, system components, or prototypes. Note that the C4ISR component can also include an embedded simulation within it, used for COA analysis or other forms of prediction. The simulation and C4ISR components communicate with one another in both directions. The states of friendly entities, and of visible opposing entities, and the effects of any relevant scenario events, are reported by the simulation to the C4ISR system, via various modeled sensor and communication systems. As the C4ISR system issues commands, they are passed back to the simulated entities that are to carry them out. In the center, visualizing the state of the simulation, as well as the plans and perceptions of the C4ISR system, either separately or together is the Integrated Situation Viewer application.

At the top, the Evaluation component provides a collection of tools for analyzing data produced by both the simulation and C4ISR systems. This component addresses the post-execution aspects of a simulation experiment. It remains the least developed part of the JSB-RD environment.

2.2 JSB-RD Components

The components of the JSB-RD distributed simulation environment are shown in Figure 2-2.

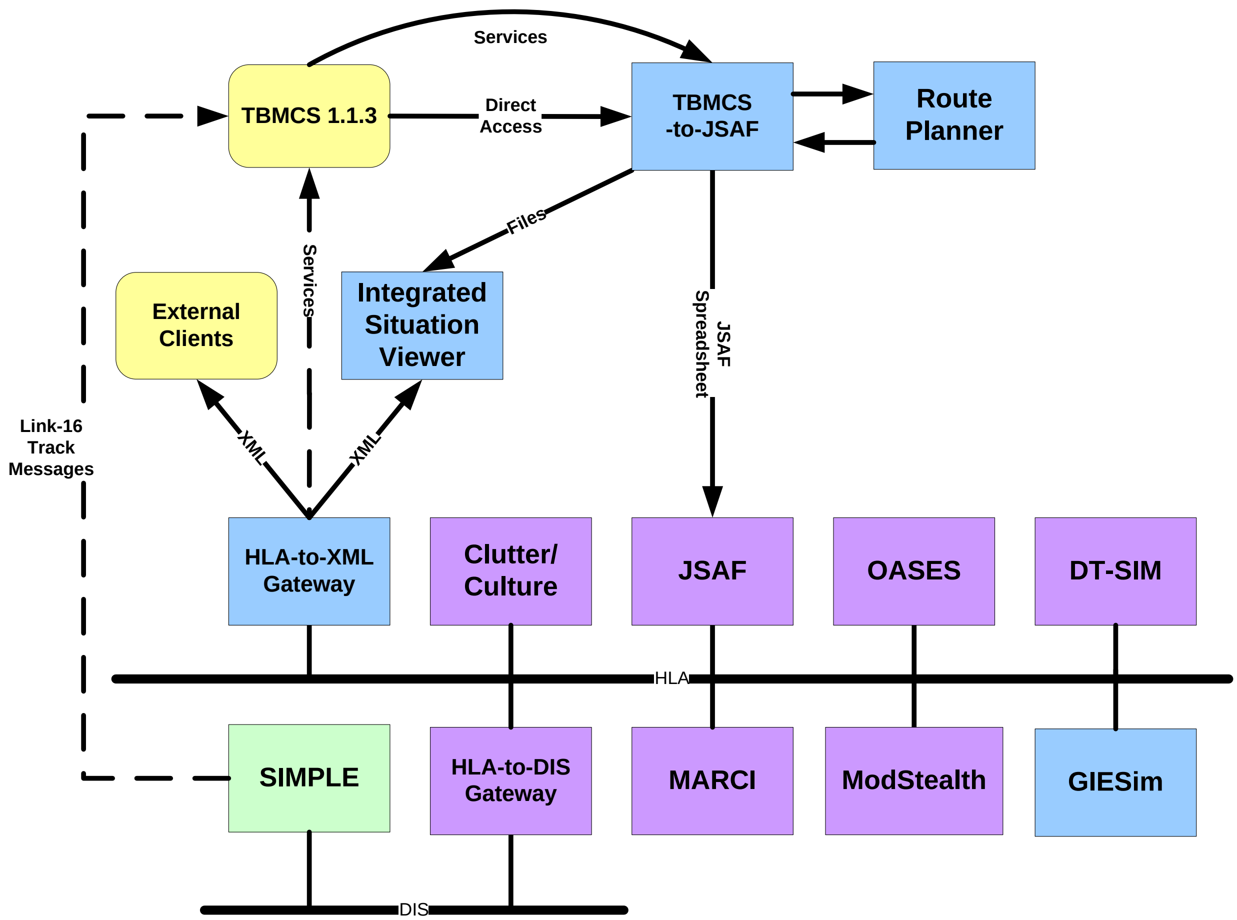


Figure 2-2: JSB-RD Components

JSAF, OASES, GIESim, and several supporting simulations make up an HLA federation. The HLA-to-XML Gateway translates the federation message traffic into an XML stream that can be easily read by a variety of applications. One such application is the Integrated Situation Viewer, which receives and displays entity state and interaction information. The Simulation Preparation tool extracts Air Battle Plan and Friendly Order of Battle information from the AODB, and Enemy Order of Battle information from the MIDB, which it then uses to generate a scenario input spreadsheet that can be read by JSAF to create and task the necessary simulation entities. The Simulation Preparation tool also interfaces with an aircraft route-planning tool created at AFRL to generate more realistic air mission flight paths that avoid known threats.

2.2.1 JSAF

JSAF is currently the primary simulation used within the JSB-RD environment. As noted above, JSAF is an entity-level, computer generated forces (CGF) simulation system that is used by the U.S. Joint Forces Command for joint experimentation, by the U.S. Navy for Fleet Battle Experiments, and by the AFRL Human Effectiveness Directorate (AFRL/HE) in support of the Distributed Mission Training (DMT) program. JSAF provides entity-level simulation of ground, air, and naval forces. In support of the joint experiment Urban Resolve, it has been used by JFCOM to simulate more than 100,000 entities within a single distributed simulation. It has also been used to support a variety of experiments with environment simulation, including dynamic terrain (i.e., craters, trenches, etc.), weather, and chemical/biological warfare defense, as part of DMSO's EnviroFed program. JSAF was originally developed by DARPA as part of its Synthetic Theater of War (STOW) program, and is descended from ModSAF. JSAF is maintained by the Joint Forces Command (JFCOM). It incorporates intelligent agents, due to the Soar program, that autonomously controls simulated fixed wing aircraft.

As part of this effort, the existing JSAF installation was updated to JSAF 2004. The FOM was updated to the Joint Urban Operations (JUO) FOM. Detailed urban databases of Jakarta, Indonesia and Baghdad, Iraq, each containing thousands of buildings, were obtained and installed.

JSAF is an extremely large software application. It was originally developed more than 15 years ago, in C, and has been extensively modified and extended. As a result, its original architecture has been almost completely obscured. It now consists of more than 1000 object libraries that model different types of platforms, weapons, sensors, etc. While it contains a great deal of powerful simulation functionality, it is difficult to use, and even more difficult to modify. Documentation and support are both extremely limited.

JSAF runs under the Linux operating system. In the JSB-RD environment, JSAF can be run on multiple Linux systems simultaneously. The individual copies function together as a single HLA federate, keeping their separate internal database copies in synchronization, and sharing the computational load.

The primary input to JSAF is a spreadsheet file that defines a collection of entities, both friendly and enemy, to be created, and specifies how each entity is to be tasked. Entity types, initial locations, call signs, and assigned tasks are identified. Such spreadsheets can be prepared by hand. However, they can also be automatically generated from Air Battle Plan information, supported by Friendly and Enemy Order of Battle information, extracted from the AODB and MIDB within TBMCS.

JSAF scenarios can also be created interactively, using its integral Unit Editor. Individual entities and small units can be created, placed on the Plan View Display (PVD), and assigned tasks to perform. Interactively created scenarios can be saved and (re)loaded. However, such scenarios and the scenario spreadsheets are two separate mechanisms, and cannot be readily combined.

JSAF is capable of receiving and processing several types of information via HLA. This includes entity state information output by other simulations within the federation. For example, JSAF reads the states of civilian vehicles and pedestrians that are modeled by the clutter simulation, and displays them on the JSAF Plan View Display. Any such external entities are visible to the JSAF-controlled entities; they can be detected, fired at, collided with, etc. JSAF also reads the weather state information output by OASES, although most platform and equipment models within JSAF do not make use of such information. JSAF also reads messages describing changes to the terrain that are output by DTSim. Such dynamic terrain changes (e.g., the appearance of a bomb crater) can affect the movement of ground vehicles in JSAF.

The primary output of JSAF is entity state information for all of the entities that it models. It also outputs various types of interactions, including weapons fire and detonation, collision, etc. In support of various exercises and experiments, JSAF has been extensively modified to support additional FOMs. It includes an extensive FOM agility layer. Many of its outputs are platform- or weapon-system specific.

Figure 2-3 shows the JSAF PVD, with a map background (in this case showing part of Baghdad) and platform-level icons. The toolbar at the upper left provides access to a variety of tools for creating, tasking, and otherwise manipulating the simulated entities. Multiple copies of the JSAF PVD can be run simultaneously. Commonly, some JSAF GUIs are congifured as controller workstations, which see and can manipulate all entities, while others are configured as "player" workstations, which can see only their own forces, as well as any enemy forces that have been detected.

The JSB-RD simulation environment currently uses a version of the Joint Urban Operations (JUO) federation object model (FOM). The JUO FOM was developed by JFCOM for use in the Joint Urban Resolve exercise. It is based on the Realtime Platform Reference (RPR) FOM, which was designed to map the original DIS protocols into an HLA environment. The JUO FOM includes a number of objects and interactions that were added to support specific aspects of the Millenium Challenge 2002 (MC02) and Joint Urban Resolve exercises. Most of these are not currently used within the JSB-RD environment. The JUO FOM elements that are currently used include:

- the BaseEntity/PhysicalEntity/Platform class, primarily the GroundVehicle and Aircraft subclasses,
- the Atmosphere, SurfaceWeather, and Weather classes, and their various subclasses,
- the Collision, Weapon Fire, and Munition Detonation interactions, and
- the GIESim Entity State, Message Send, and Message Received interactions.

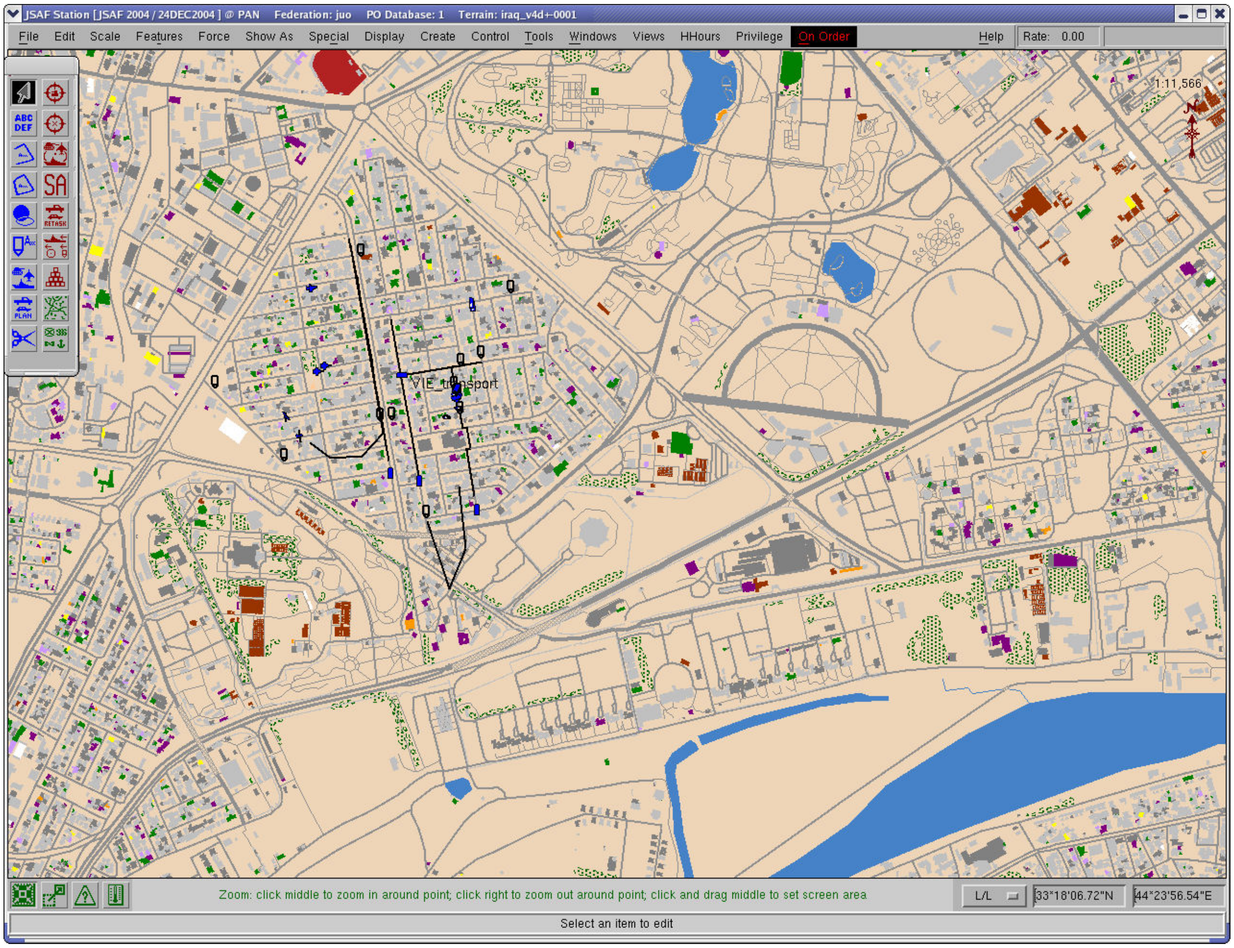


Figure 2-3: JSAF Plan View Display with Air Dynamics Editor

2.2.2 OASES

The OASES system is a suite of applications for creating and managing a three-dimensional, time-varying, digital representation of the natural environment. OASES has been used primarily to provide synthetic natural environments (SNEs) to systems of networked military training simulations running on the Department of Defense's (DoD) High Level Architecture (HLA). The simulated natural environments created by OASES are based on authoritative, validated numerical models, typically the same models that are used by METeorological/OCeanographic (METOC) personnel in support of real-world military operations. OASES provides tools for converting authoritative model outputs to a data format recognized by all of the OASES applications. This format supports the data access requirements of distributed simulations that integrate virtual and/or live entities and which must operate in real-time. Additionally, OASES provides tools for tailoring the SNE, either before the simulation begins or while it is running, to meet exercise-specific requirements for environmental phenomena.

Under this effort, the existing OASES installation was updated to the most recent version.

The original development of the OASES system was sponsored by the Defense Advanced Research Project Agency (DARPA), under the name *TAOS* (Total Atmosphere Ocean Services), in support of the Synthetic Theater of War (STOW) 1997 Program. In 1998, DARPA funded the development of a low-resolution worldwide atmospheric and oceanographic database, also known as the *Global-98* database, for use by the JSIMS program. In 1999, the United States Space Command's (USSPACECOM) Space Warfare Center funded extensions to TAOS to support the space environment, specifically ionospheric effects on precision-guided missiles, as part of the PSM+ (extended Portable Space Model) project. More recently, funding for continued development and integration with the HLA, under the name OASES, has been provided primarily by DMSO through the Environment Federation (EnviroFed) projects.

Figure 2-4 shows the OASES subsystems and the data flows among them. OASES converts current and forecast environmental data, provided by physics-based operational and research models, into a form that can be used by distributed simulations running on the HLA.

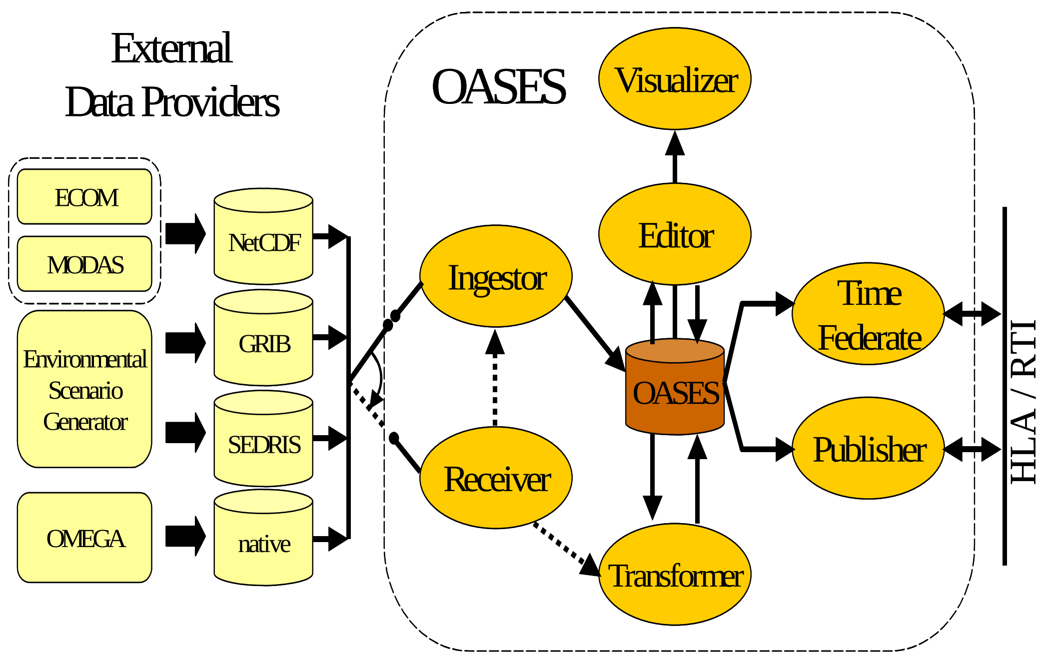


Figure 2-4: OASES Organization

The OASES system consists of five primary subsystems. The OASES Ingestor converts all input model data to a common run-time format that is recognized by all OASES subsystems. The external data providers are identified in the rounded-rectangles on the left side of the figure. The OASES Transformer uses a set of transformation algorithms to augment existing OASES databases with various environmental parameters that are not provided directly by an external data source, but that are required by the simulations served by OASES. The OASES Editor allows users to tailor the contents of an OASES database. The Editor provides three editing algorithms: 1) replacement at a point with gaussian spatial and temporal blending, 2) a Pressure Field Modification (PFM) algorithm for editing atmospheric environments while preserving correlation between temperature, pressure, wind and relative humidity, and 3) a precipitation editing algorithm. The Editor can be used to prepare scripted changes to an existing METOC scenario, or it can be used at run-time to modify the SNE during a simulation exercise. The OASES Time Federate and Publisher are the subsystems that interface directly to the HLA Federation; that is, they are the OASES *Federates*. They create and update the objects that encapsulate the state of the simulated natural environment via services provided by the RTI. The Time Federate creates objects that establish the time-dependence of the SNE while the Publisher manages objects that encapsulate its spatial-dependence.

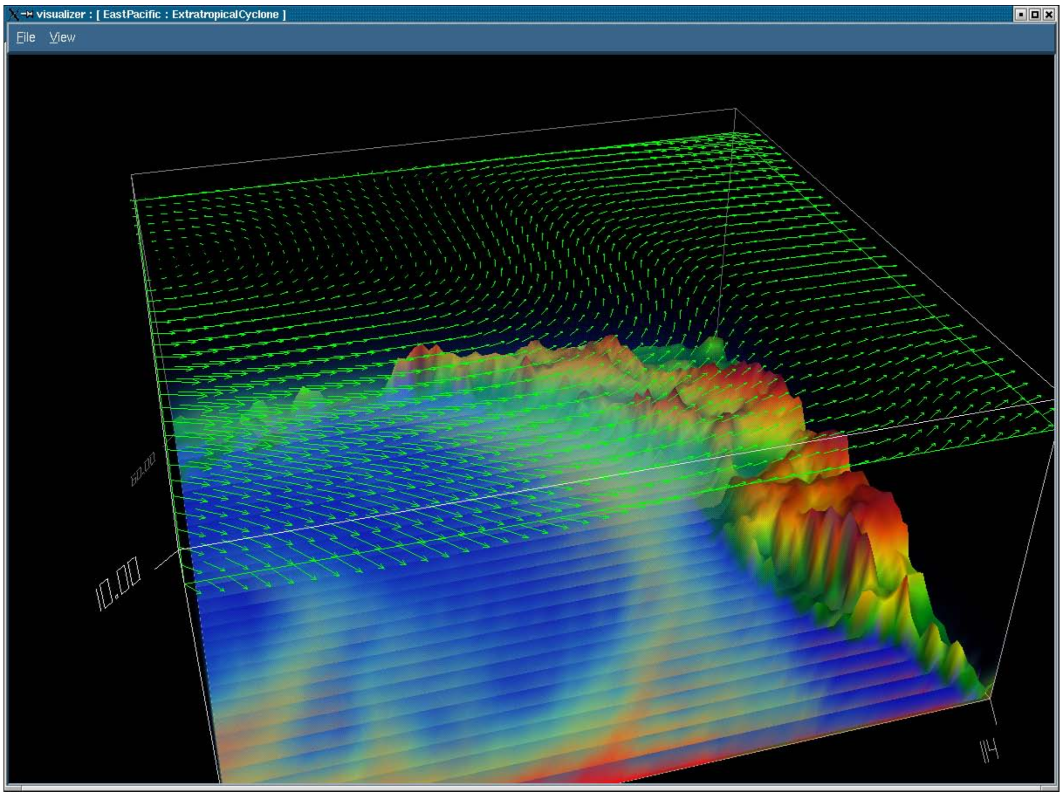


Figure 2-5: OASES Weather Visualization

The OASES Visualizer is a tool for visualizing the contents of an OASES database. It is used to validate databases built by the Ingestor and/or extended by the Transformer, to review the results of edits applied by the OASES Editor, to monitor the current state of the SNE created by the Publisher, and/or to monitor the current state of the SNE as received by the OASES Subscriber. An example of an OASES Visualizer display is shown in Figure 2-5.

Finally, the OASES Receiver is responsible for polling local or remote data sources, using the Internet File Transfer Protocol (FTP), for environmental data transmittals matching a user-specified file-naming pattern. The Receiver is the subsystem that supports the“Live Mode” of OASES, in which the simulated natural environment is continuously updated based on the data received from current and forecast environmental models running in real-time.

Within the JSB-RD environment, OASES is used to bring in weather information from Air Force and Navy sources. OASES outputs weather state information, including temperature, pressure, and precipitation information, in one-dimensional (profile), two-dimensional (surface), and three-dimensional forms, over HLA. This information is read by JSAF and DTSim, which use it to modify some military operations, and to implement changes to the terrain database, respectively.

2.2.3 Culture/Clutter Simulation

The Culture/Clutter simulation, which is part of the JSAF distribution, models the movements of civilian vehicles and pedestrians. The amount and type of clutter is specified using clutter templates. Each template specifies a list of entity types with associated relative weights, as well as a collection of control points. Each control point specifies a center location, a radius (defining a circular area), and a number of clutter entities. Each control point is identified as defining a static clutter area, a mobile clutter area, a clutter source, or a clutter sink. Clutter entities move randomly with a static clutter area. Sources and sinks allow dynamic traffic flows to be created. Clutter etities are randomly created in the source areas, and move to random locations within a sink area. When they arrive, they are destroyed and replaced with a new entity in one of the source areas. The clutter simulation publishes entity state information for each of the clutter entities as they move.

In support of the JFCOM Urban Resolve joint experiment, the Culture/Clutter simulation has been significantly enhanced. A wide variety of clutter entities, including both vehicles and lifeforms, can be created. Templates can be defined that specify the movements of clutter entities at specific times. These templates can be used to more realistically model civilian traffic movements such as commuting to and from work, political demonstrations, etc.

2.2.4 DTSim

The Dynamic Terrain Simulation (DTSim) models changes to the terrain component of the environment. The changes result from various types of simulation events, including weapon detonations, movement, weather effects, and military engineering operations, such as the creation and destruction of obstacles. DTSim receives interaction events from JSAF, and determines what effect, if any, they have on the geometry or attributes of the terrain at the location event. For example, when DTSim receives a weapon detonation interaction from JSAF, it may, depending on the type of the weapon and its proximity to the terrain surface, or a specific terrain feature, determine that a crater should be created, or that a building should be damaged. DTSim updates its internal terrain representation, and publishes messages over HLA describing how the terrain has changed. These messages are used by JSAF to update its terrain representation, which in turn affects how some activities are carried out. For example, the appearance of a new crater may change the movement of vehicles to avoid it. Weather effects, such as prolonged rain, may also change the characteristics of the terrain, restricting the speed of cross-country movement.

2.2.5 ModStealth

ModStealth is a 3D “stealth” viewer that is part of the JSAF software distribution. It provides three-dimensional perspective views of the scenario entities and environment that are dynamically updated as entities move and events occur. Figure 2-6 shows an example perspective view of part of the Baghdad urban database. The viewpoint can either be attached to a specified scenario entity, or can be manually manipulated. The ModStealth control panel is shown in Figure 2-7.

ModStealth was used extensively to capture screen footage to support the visualization of the DARPA DNCW concept. This exposed several issues, as described in section 3.2. In particular, ModStealth requires a specialized database format that is different from the CTDB format used by JSAF. The extremely large and detailed Baghdad database that was used significantly stressed ModStealth's memory management capabilities.

Figure 2-6: ModStealh Perspective View

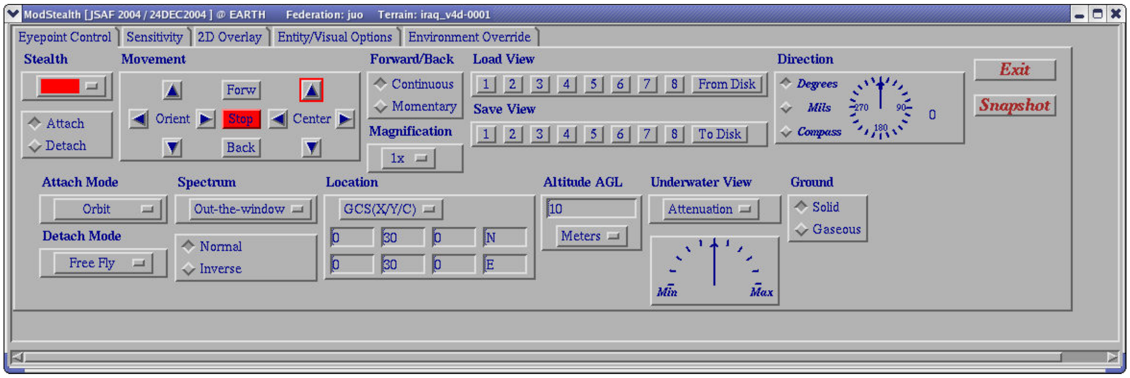


Figure 2-7: ModStealth Control Panel

2.2.6 MARCI

MARCI (Multi-host Automation Remote Control and Instrumentation) is a simulation exercise control and management tool that is part of the JSAF software distribution. MARCI allows an operator to control, monitor, and analyze an entire federation, possibly distributed across multiple sites, from a single workstation. MARCI can be used to distribute federates to multiple systems prior to a simulation run. MARCI's Mass Launch capability provides the ability to start multiple systems nearly instantaneously. MARCI can also display disk space and memory utilization on these systems. Operators have the ability to launch individual workstations by choosing from a list of common options in a graphical user interface (GUI). Individual federates can be started and shut down. MARCI can execute federation-wide Global Pause and Global Resume operations through the RTI. It can execute federation-wide scenario load and save operations.

2.2.7 HLA-to-DIS Gateway

The HLA-to-DIS Gateway is part of the JSAF software distribution. It is an HLA federate that translates a subset of the MC02 FOM messages, the subset that matches the Realtime Platform Reference (RPR) FOM, to and from corresponding DIS Protocol Data Units (PDUs). This allows DIS applications to participate in HLA federations.

2.2.8 SIMPLE

SIMPLE (Simulation to C4I Interchange Module for Plans Logistics and Exercises) is an interface between the simulated battlefield environment and real world command and control systems. SIMPLE provides a database that maps simulation units, platforms, munitions, and supplies to real world units, platforms, munitions and supplies.

SIMPLE also contains a messaging module that correctly generates the tactical messages required by the military C4I systems to report on these units, platforms, etc.

The heart of SIMPLE is the scenario database. This database is uniquely tailored to each simulation scenario in order to provide the correct mappings from "sim" to reality. SIMPLE is a product in the Digital Battlestaff Sustainment Training (DBST) federation environment developed primarily by the National Simulation Center (NSC) located at Ft. Leavenworth, KS.

SIMPLE was used to generate air track messages in Link-16 format, for input to the TBMCS Track Management Data Base (TMDB). This is discussed in detail in section 3.2.

2.2.9 HLA-to-XML Gateway

The HLA-to-XML Gateway is an HLA federate that translates FOM entity and interaction messages into an easily parsable XML stream that can be accessed by multiple applications. The intent of the HLA-to-XML Gateway is to make it easier for a variety of applications to access the data generated by an HLA federation, by encapsulating the details of connecting to an HLA federation and receiving data.

Under this effort, the HLA-to-XML Gateway software was enhanced to support HLA interactions, and to calculate the orientations of aircraft and other moving platforms.

Currently, the HLA-to-XML Gateway is primarily used to support the Integrated Situation Viewer.

2.2.10 Integrated Situation Viewer

The Integrated Situation Viewer, commonly referred to simply as “the Viewer”, is intended to support experimentation with theater-level air mission situation awareness and dynamic retasking. The Viewer displays simulation state information describing air missions, which is, in effect, ground truth, and the corresponding Air Battle Plan information. In addition, any information (derived from the simulation) that reflects the reported or perceived current situation, as output by various sensor models, is displayed. In its current form, the Viewer consists of a number of loosely coupled components. Figure 2-8 shows the architecture for the Viewer, which is based on the classic Model-View-Controller paradigm.

The back-end portion of the Viewer consists of a collection of components that are capable of reading data from various sources. Simulation entity state and event data output by JSAF, OASES, DTSim, and other simulations is read via an HLA-to-XML gateway. This gateway converts entity state information received over HLA into a stream of XML messages. It also performs coordinate conversion (from geocentric coordinates to geodetic coordinates), and partial translation of the DIS Entity Bit Vector (EBV) fields that are used to hierarchically identify entities by nationality, domain, type, subtype, etc. Other data sources, including JSAF input spreadsheets, are read directly from the appropriate files.

The “heart” of the Viewer consists of an integrated domain model that stores and maintains information on air mission plans, individual aircraft, and their targets. The data read from the various back-end sources is used to populate and update this model.

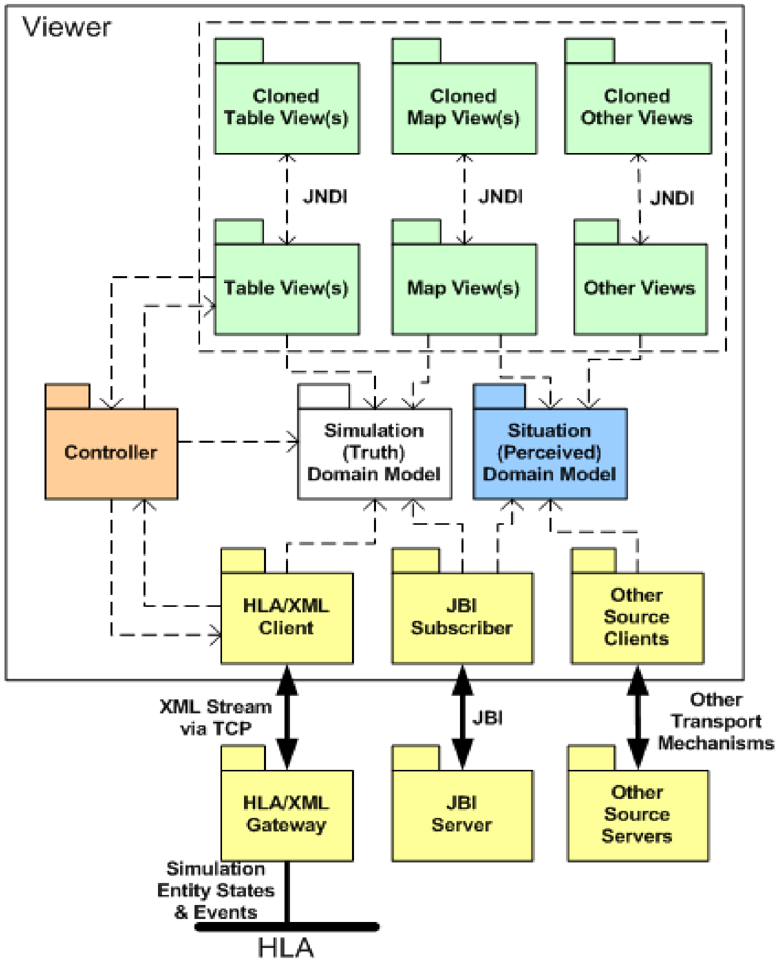


Figure 2-8: Viewer Architecture

The front-end of the Viewer consists of multiple views of the information that the domain model contains. Several types of views are supported, including:

- Map views – displaying the locations of aircraft and targets, planned and actual flight paths, etc., overlaid on a map background at multiple scales and resolutions using the JView visualization toolkit developed by AFRL/IFSB (see Figure 2-9),
- Tabluar views – listing entities of various types (aircraft, targets, missions, etc.) and their relevant characteristics, and capable of being sorted and ordered in various ways,
- Text views – displaying a streaming list of text messages, generated by the Simulation Network News (SNN) utility, which is part of the JSAF distribution.

Under this effort, an interactive retasking capability was added to the viewer. When this capability is invoked, it sends a message to JSAF to alter the current tasking of a specified aircraft, normally designating a new target for that aircraft.

The main Viewer GUI, shown in Figure 2-9, consists of a default Map View and a row of buttons at the bottom of the window to start other views. With the exception of the SNN View, multiple instances of all of the views can be launched.

Figure 2-9: Main (Map View) GUI

The Map View contains entities, routes and background map data. It is also where the lens appears (see Figure 2-10).

The lens feature allows the user to left click anywhere in the main GUI, causing both a lens and a separate magnified Map View to appear. The lens can be moved by clicking and holding the mouse button within the title bar region of the lens window. Clicking in another location on the map will cause the lens to snap to that location. The movement of the lens is reflected by the map information in the Magnified Map View.

Figure 2-10: Lens View

The Magnified Map View, shown in Figure 2-11, shows the same data as the Map View but with both the CADRG and DTED map background at a higher resolution than the Map View. The center divider can be moved to adjust the amount of space both map types take up on the screen. Any entity icon selections made in the Magnified Map View will result in the selected item being highlighted in the Table View if it is visible there. Route selections made in the Magnified Map View will result in the Route Name being output to the command window.

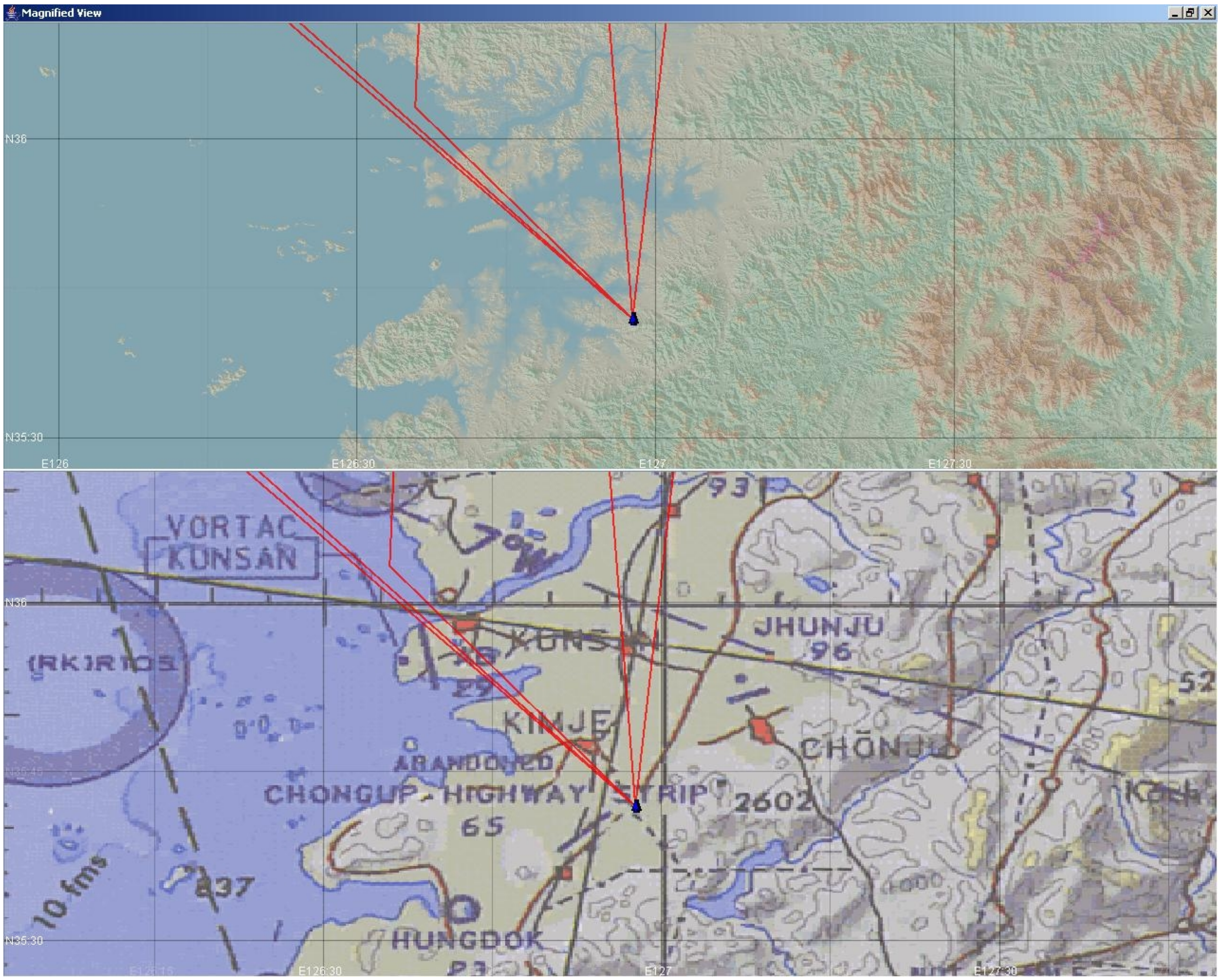
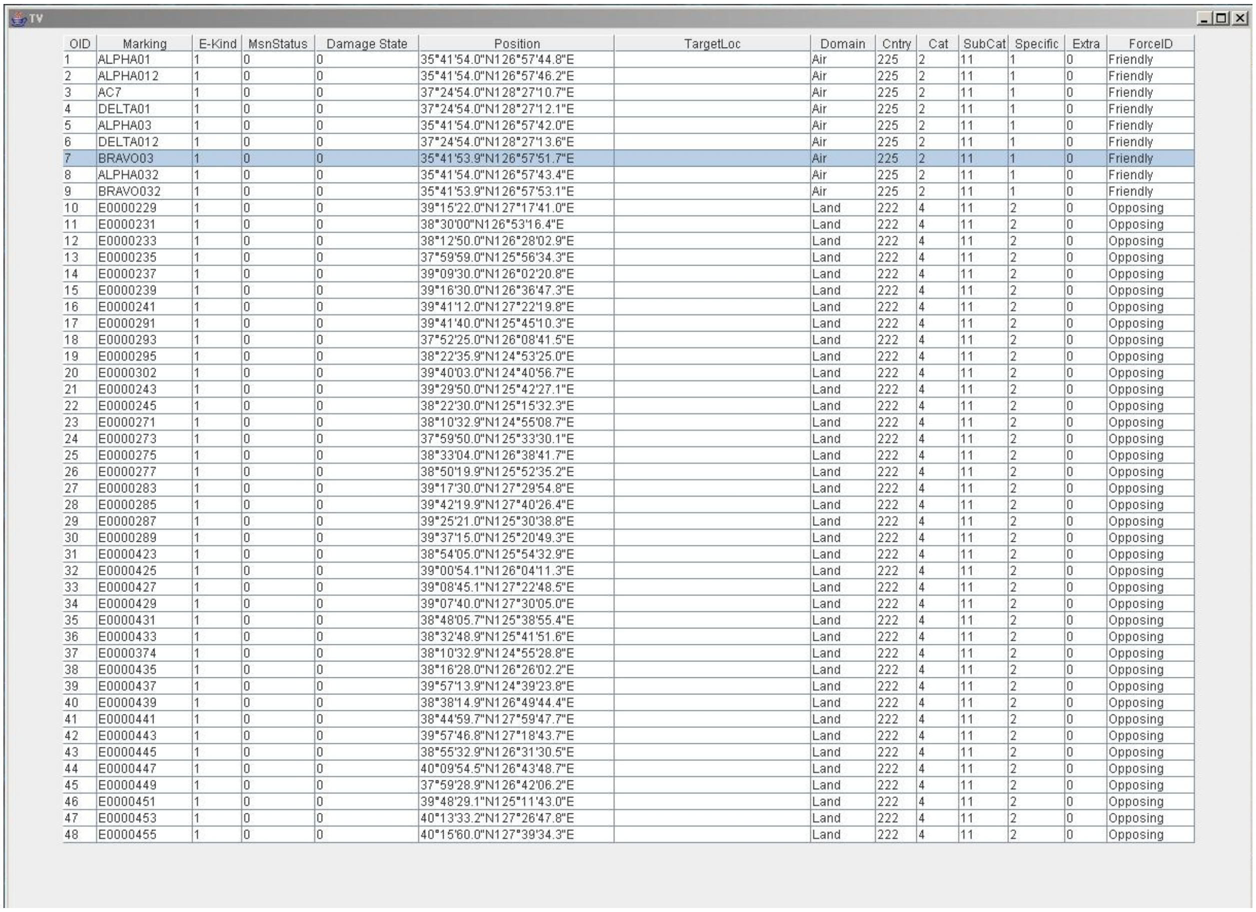


Figure 2-11: Magnified Map View

The Table View, shown in Figure 2-12, is launched via the New Table View button at the bottom of the main GUI window. This view contains data related to the entities shown on the map. The columns can be moved around and the rows can be sorted by clicking on the column header. Complex sorts are enabled by holding the control key down when clicking subsequent column headers. A row selected in the table causes the related entity icon to be highlighted on the Map View.



OID	Marking	E-Kind	MsnStatus	Damage State	Position	TargetLoc	Domain	Cntry	Cat	SubCat	Specific	Extra	ForceID
1	ALPHA01	1	0	0	35°41'54.0"N126°57'44.8"E		Air	225	2	11	1	0	Friendly
2	ALPHA012	1	0	0	35°41'54.0"N126°57'46.2"E		Air	225	2	11	1	0	Friendly
3	AC7	1	0	0	37°24'54.0"N128°27'10.7"E		Air	225	2	11	1	0	Friendly
4	DELTA01	1	0	0	37°24'54.0"N128°27'12.1"E		Air	225	2	11	1	0	Friendly
5	ALPHA03	1	0	0	35°41'54.0"N126°57'42.0"E		Air	225	2	11	1	0	Friendly
6	DELTA012	1	0	0	37°24'54.0"N128°27'13.6"E		Air	225	2	11	1	0	Friendly
7	BRAVO03	1	0	0	35°41'53.9"N126°57'51.7"E		Air	225	2	11	1	0	Friendly
8	ALPHA032	1	0	0	35°41'54.0"N126°57'43.4"E		Air	225	2	11	1	0	Friendly
9	BRAVO032	1	0	0	35°41'53.9"N126°57'53.1"E		Air	225	2	11	1	0	Friendly
10	E0000229	1	0	0	39°15'22.0"N127°17'41.0"E		Land	222	4	11	2	0	Opposing
11	E0000231	1	0	0	38°30'00"N126°53'16.4"E		Land	222	4	11	2	0	Opposing
12	E0000233	1	0	0	38°12'50.0"N126°28'02.9"E		Land	222	4	11	2	0	Opposing
13	E0000235	1	0	0	37°59'59.0"N125°56'34.3"E		Land	222	4	11	2	0	Opposing
14	E0000237	1	0	0	39°09'30.0"N126°02'20.8"E		Land	222	4	11	2	0	Opposing
15	E0000239	1	0	0	39°16'30.0"N126°36'47.3"E		Land	222	4	11	2	0	Opposing
16	E0000241	1	0	0	39°41'12.0"N127°22'19.8"E		Land	222	4	11	2	0	Opposing
17	E0000291	1	0	0	39°41'40.0"N125°45'10.3"E		Land	222	4	11	2	0	Opposing
18	E0000293	1	0	0	37°52'25.0"N126°08'41.5"E		Land	222	4	11	2	0	Opposing
19	E0000295	1	0	0	38°22'35.9"N124°53'25.0"E		Land	222	4	11	2	0	Opposing
20	E0000302	1	0	0	39°40'03.0"N124°40'56.7"E		Land	222	4	11	2	0	Opposing
21	E0000243	1	0	0	39°29'50.0"N125°42'27.1"E		Land	222	4	11	2	0	Opposing
22	E0000245	1	0	0	38°22'30.0"N125°15'32.3"E		Land	222	4	11	2	0	Opposing
23	E0000271	1	0	0	38°10'32.9"N124°55'08.7"E		Land	222	4	11	2	0	Opposing
24	E0000273	1	0	0	37°59'50.0"N125°33'30.1"E		Land	222	4	11	2	0	Opposing
25	E0000275	1	0	0	38°33'04.0"N126°38'41.7"E		Land	222	4	11	2	0	Opposing
26	E0000277	1	0	0	38°50'19.9"N125°52'35.2"E		Land	222	4	11	2	0	Opposing
27	E0000283	1	0	0	39°17'30.0"N127°29'54.8"E		Land	222	4	11	2	0	Opposing
28	E0000285	1	0	0	39°42'19.9"N127°40'26.4"E		Land	222	4	11	2	0	Opposing
29	E0000287	1	0	0	39°25'21.0"N125°30'38.8"E		Land	222	4	11	2	0	Opposing
30	E0000289	1	0	0	39°37'15.0"N125°20'49.3"E		Land	222	4	11	2	0	Opposing
31	E0000423	1	0	0	38°54'05.0"N125°54'32.9"E		Land	222	4	11	2	0	Opposing
32	E0000425	1	0	0	39°00'54.1"N126°04'11.3"E		Land	222	4	11	2	0	Opposing
33	E0000427	1	0	0	39°08'45.1"N127°22'48.5"E		Land	222	4	11	2	0	Opposing
34	E0000429	1	0	0	39°07'40.0"N127°30'05.0"E		Land	222	4	11	2	0	Opposing
35	E0000431	1	0	0	38°48'05.7"N125°38'55.4"E		Land	222	4	11	2	0	Opposing
36	E0000433	1	0	0	38°32'48.9"N125°41'51.6"E		Land	222	4	11	2	0	Opposing
37	E0000374	1	0	0	38°10'32.9"N124°55'28.8"E		Land	222	4	11	2	0	Opposing
38	E0000435	1	0	0	38°16'28.0"N126°26'02.2"E		Land	222	4	11	2	0	Opposing
39	E0000437	1	0	0	39°57'13.9"N124°39'23.8"E		Land	222	4	11	2	0	Opposing
40	E0000439	1	0	0	38°38'14.9"N126°49'44.4"E		Land	222	4	11	2	0	Opposing
41	E0000441	1	0	0	38°44'59.7"N127°59'47.7"E		Land	222	4	11	2	0	Opposing
42	E0000443	1	0	0	39°57'46.8"N127°18'43.7"E		Land	222	4	11	2	0	Opposing
43	E0000445	1	0	0	38°55'32.9"N126°31'30.5"E		Land	222	4	11	2	0	Opposing
44	E0000447	1	0	0	40°09'54.5"N126°43'48.7"E		Land	222	4	11	2	0	Opposing
45	E0000449	1	0	0	37°59'28.9"N126°42'06.2"E		Land	222	4	11	2	0	Opposing
46	E0000451	1	0	0	39°48'29.1"N125°11'43.0"E		Land	222	4	11	2	0	Opposing
47	E0000453	1	0	0	40°13'33.2"N127°26'47.8"E		Land	222	4	11	2	0	Opposing
48	E0000455	1	0	0	40°15'60.0"N127°39'34.3"E		Land	222	4	11	2	0	Opposing

Figure 2-12: Table View

The Controller is the “brain” of the Viewer, coordinating all of the other components. When new information arrives from one of the sources, the Controller triggers the updating of the domain model, and then the updating of all views that are affected by the change. Similarly, when the interactively changes a piece of information in one of the views, the Controller triggers the updating of the domain model, and then the updating of any other views that are affected by the change. The Controller also coordinates the selection of entities and locations across all of the views, so that an entity that is selected in one of the views is also highlighted in all other views in which it appears. A Controller GUI allows the user to select which types of views should be displayed and what the content of each should include.

A planned enhancement to the Viewer was to make use of a custom implementation of the Java Naming and Directory Interface (JNDI), developed by AFRL/IFSB, to allow views to be "cloned" across multiple workstations and monitors. However, this JNDI implementation was never completed.

2.2.11 TBMCS-to-JSAF

The simulation preparation component of the JSB-RD environment, TBMCS-to-JSAF, allows existing Air Battle Plans (ABPs) contained within the Air Operations Data Base (AODB) of the Theater Battle Management Core System (TBMCS) to be converted into sets of JSAF input spreadsheets that can be executed using JSAF. This application extracts a specified ABP from the AODB, which contains specifications of multiple air missions of various types. The ABP selection window is shown in Figure 2-13. Each mission specification includes the numbers and types of aircraft involved in the mission, their takeoff and return times and bases, and a sequence of key mission events. These mission events include takeoff, refueling (start and end), time on target (start and end), and landing. Ground attack missions also identify their respective targets. Supporting information describing the mission targets is extracted from the MIDB.

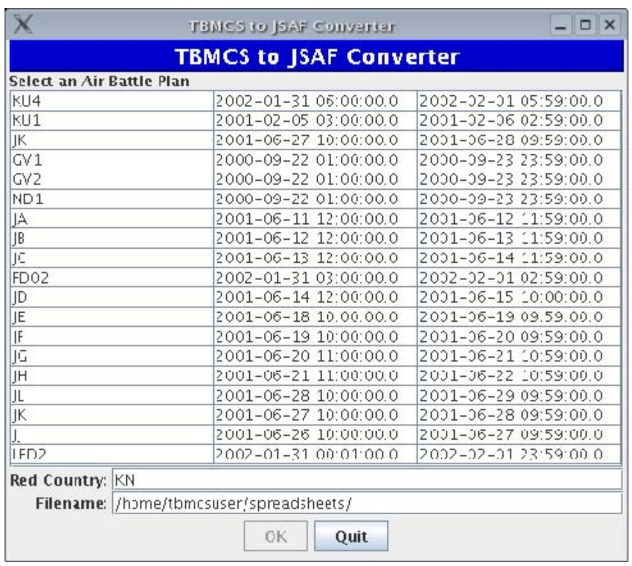


Figure 2-13 TBMCS-to-JSAF ABP Selection Window

Under this effort, TBMCS-to-JSAF was updated to interface with the current version of TBMCS, version 1.1.3, using a JDBC interface.

The basic mission information, along with a list of the air defense threats in the area, can be fed to a separate Route Planner application (see below), which determines the "best" route for each mission to and from its assigned target while avoiding air defense threats. The returned route contains a number of intermediate waypoints that the aircraft should pass through on the way to and from their target.

This information is then used to generate a JSAF input spreadsheet. The spreadsheet contains two entries for each scheduled air mission, one describing the ingressing leg of the mission, and the other describing the return leg. Each entry specifies, in JSAF terms, the number and type of aircraft, the call sign(s) of the aircraft, the type of task to be performed, the take off and return times, and the base and target locations. A second spreadsheet contains the intermediate route points, which are linked to the missions by name.

2.2.12 Route Planner

The Route Planner application takes a “stick route” for a planned air mission, consisting only of the source airbase coordinates, the target coordinates, and the return airbase coordinates, as well as a collection of adversary air defense threats (i.e., SAM and AAA sites). It returns a more complex route, containing additional waypoints, that attempts to reach the target and return while avoiding the listed threats.

The Route Planner is used by TBMCS-to-JSAF to attempt to emulate the more detailed mission planning activities that occur at the unit level. It passes the Route Planner the basic mission information obtained from an Air Battle Plan, and the air defense threats obtained from the MIDB. It takes the resulting route, with the added waypoints, and constructs a spreadsheet that can be read by JSAF.

2.2.13 GIESim

The Global Information Enterprise Simulation (GIESim) provides high fidelity Link-16 network modeling with full resolution of propagation effects, including power and distance based Signal to Noise ratio, terrain masking and other Line Of Sight (LOS) issues. The vision of GIESim is to move, process, manage, and protect the C4ISR information that supports the functions of Global Awareness and Dynamic Planning and Execution. The mission of GIE is to link aerospace assets in-theater and globally, to integrate C3 & ISR networks, to defend critical information systems from cyber attack, and to develop new information processing and management techniques. Most large-scale force level simulations assume perfect communications. The lack of communications in a simulation environment can lead to the prediction of erroneous results. Tools are needed to bridge these communications modeling gaps.

The GIESim project vision is to define, design and implement a Modeling and Simulation (M&S) framework for the Global Information Enterprise (GIE). Within the GIESim framework, users are able to execute, via a common interface, multiple communications and network M&S tools to effectively and efficiently analyze candidate communications architectures and technologies. The GIESim can interface with other M&S tools (e.g., force-level simulations and detailed hardware system models) to provide the appropriate level of M&S fidelity and processing speed for the broad spectrum of M&S tasks. The GIESim user base spans advanced technology researchers to communications network architects to mission planners.

Within the JSB-RD federation, the role of GIESim is to evaluate communications connectivity between various entities in the simulation. Communication networks are defined within GIESim, tying together various collections of simulated entities. JSAF controls the movements of the simulated entities. GIESim monitors customized entity state messages published by JSAF and updates the locations and headings of the entities. When two entities need to communicate with each other, JSAF sends a request to GIESim identifying the two entities, the type of communication, and the length of the message. GIESim then determines whether or not the specified entities can communicate, either directly or via available relays. If they can communicate, GIESim returns a response to JSAF indicating the delay before the message will arrive at its destination. JSAF then schedules the delivery of the message at the indicated time.

3. ACTIVITIES

Several activities that were performed under this effort are summarized in this section. These activities include:

- Supporting AFAMS JSAF-AWSIM comparison effort, to determine how the Air Force should participate in the Joint Urban Resolve exercise at JFCOM,
- Preparing a demonstration scenario for the Science Advisory Board's visit to AFRL,
- Preparing a visualization of DAPRA's Dynamic Network Centric Warfare (DNCW) concept,
- Generating Link-16 air track messages to interface with the TBMCS Track Management Data Base (TMDB).

Each of these activities is summarized in the following subsections. In addition, project staff provided software, data, training, and troubleshooting assistance to government and other contractor personnel using the JSB-RD simulation facility, either remotely (RAM Labs, Synergia) or via the AFRL Project Integration Center (PIC) to support other AFRL projects.

3.1 JSAF-AWSIM

In May 2005, a Working Group was formed to provide a recommendation to Air Force leadership on which air and space simulation(s) should be used to support JFCOM's Urban Resolve experiment. This experiment, set in Baghdad in 2015, has the goal of measuring the ability of the projected force at that time to isolate and control the urban battlespace (i.e., the four-block war) during stability operations following Iraqi elections. The Working Group was tasked to assess and compare the utility of JSAF versus the Air and Space Constructive Environment (ACE), which includes the Air Warfare Simulation (AWSIM) across a standard set of criteria. AFRL/IFSB, as a JSAF user, participated in this working group, along with AFAMS and other Air Force organizations.

AF/XPXC produced a list of ninety types of Air Force assets (aircraft, munitions, sensors, and other systems) expected to be available in the inventory in the 2015 timeframe. Under this effort, the capability of the current version of JSAF (i.e., JSAF 2004) to represent each of these asset types was evaluated. The JSAF data files defining entity, unit, and munition types were analyzed, and the JSAF entity and munition types corresponding to each of the required systems were identified. A spreadsheet summarizing the results was provided to the Working Group. A similar evaluation was performed relative to AWSIM by other Working Group members.

Both JSAF and AWSIM were found to be capable of modeling approximately 50% of the assets on this list. Furthermore, both simulations were found to model approximately the same 50% of the required systems. Thus, the most significant factor was determined to be that JSAF is already used by JFCOM in performing the Urban Resolve experiment. Using JSAF, at least initially, would avoid the additional costs associated with integrating AWSIM and ACE into the Urban Resolve experiment federation.

3.2 Scientific Advisory Board (SAB) Demonstration

Under this activity, an urban demonstration scenario was created in support of the Scientific Advisory Board's visit to AFRL Rome Research Site. The scenario was set in Baghdad, making use of the detailed Baghdad urban database with both JSAF and ModStealth. A group of VIPs is being moved through the city in a convoy of armored Humvees, as shown in Figure 3-1. A C-130 provides close air support of the convoy, while several UAVs and UCAVs provide ISR and possible close air support, as well as serving as communication relays. The convoy route and UAV/UCAV orbits are shown in Figure 3-2.

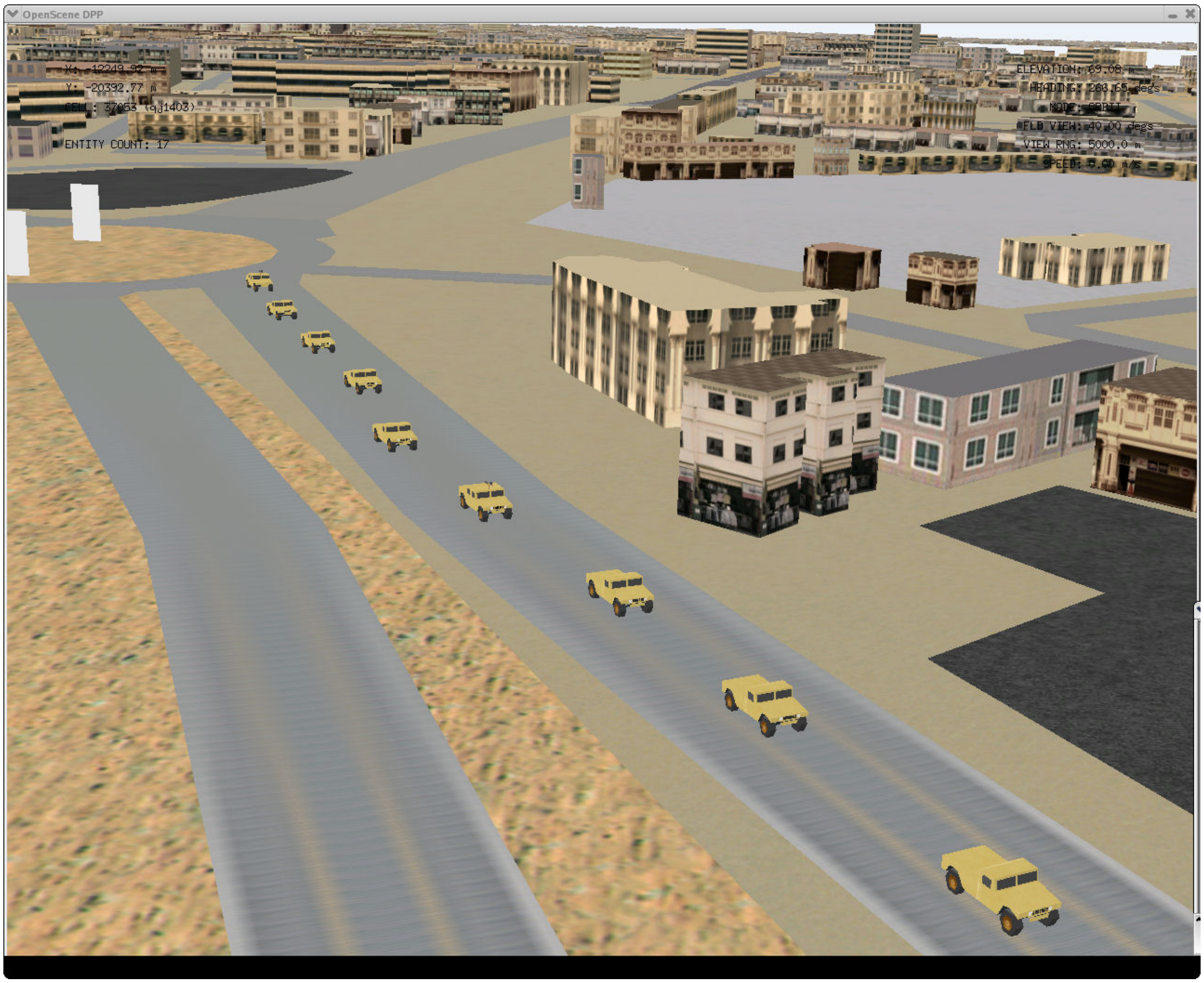


Figure 3-1. Convoy Moving Through Baghdad

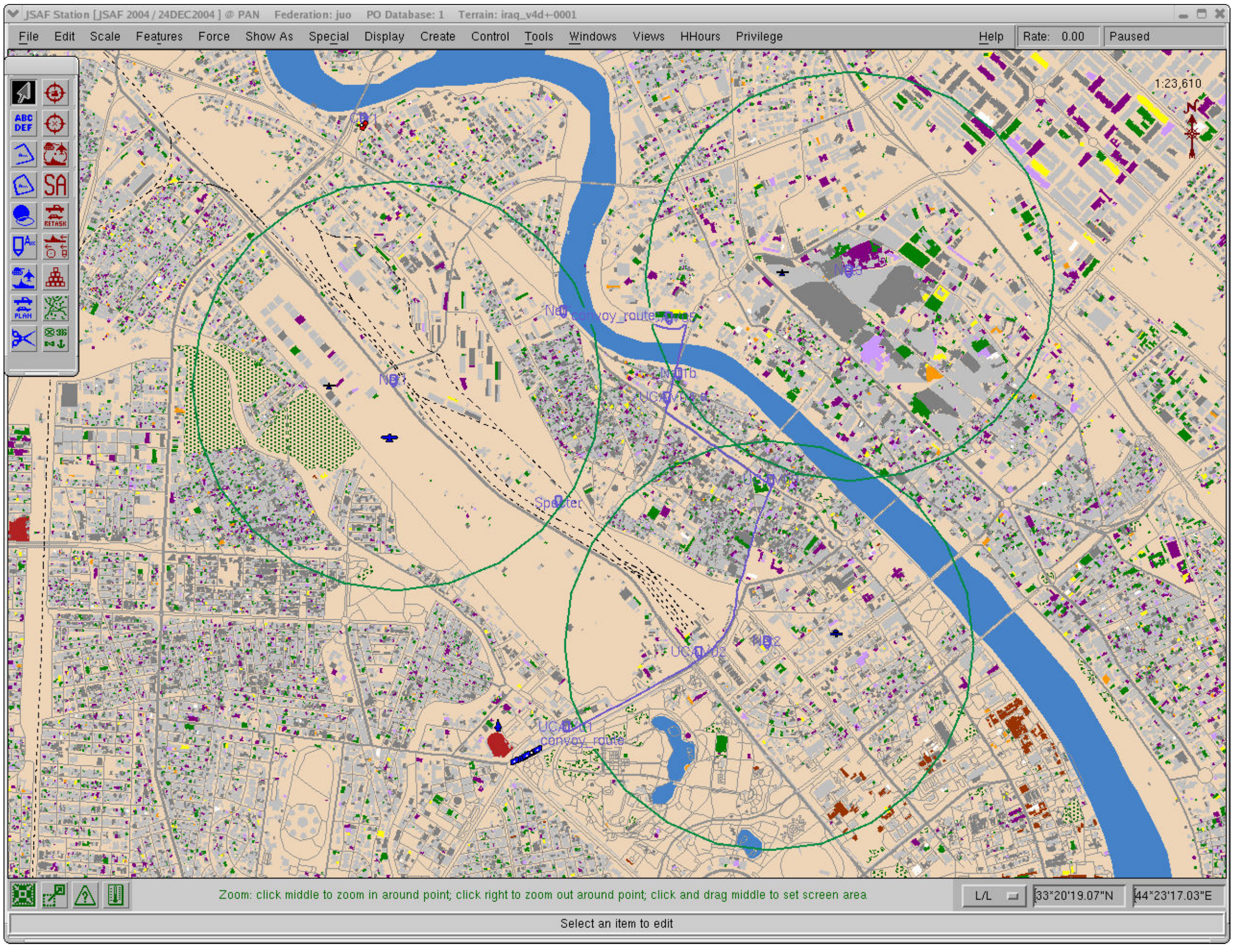


Figure 3-2. Convoy Route and UCAV Orbits

Insurgents have discovered the route and timetable of the convoy, and have planned an ambush as the convoy crosses a bridge. However, one of the UAVs, monitoring a known safe house, detects an increased level of activity as the insurgents assemble there. Several different courses of action can then be explored, including retasking one of the UCAVs to attack the safe house, retasking the convoy to alter its route, or retasking the C-130 to intercept the insurgents after the UAV has monitored their movements to the ambush site.

3.3 DNCW Visualization

This activity involved generating material for a short video detailing the Dynamic Network Centric Warfare (DNCW) concept, using animated footage captured from the ModStealth 3D visualization tool while running the JSAF simulation software. This consisted of the following sequence of steps:

1. Generating a notional scenario whose events typify aspects of the DNCW concept deemed desirable to illustrate.

Figure 3-3. Insurgent Roadblock

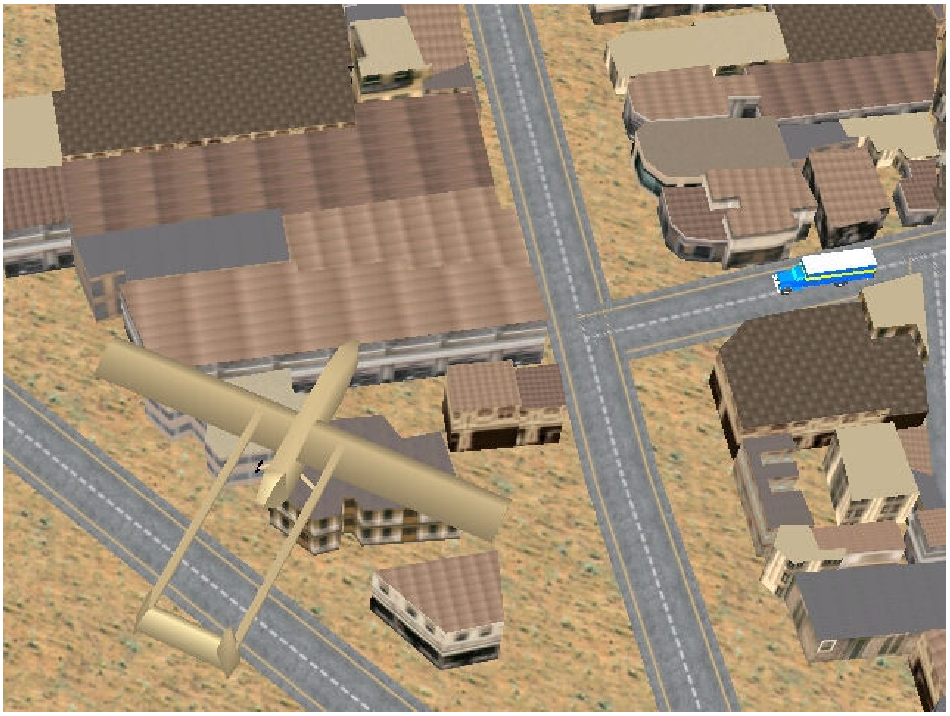


Figure 3-4. UAV Tracking Fleeing Truck

Figure 3-5. Stryker Moving to Intercept Fleeing Truck

Figure 3-6. F-16 Diverting to Attack Roadblock

2. Generating a storyboard breaking the scenario down into component scenes for filming.

3. Creating JSAF scenarios containing the desired units behaving in accordance with each desired scene.
4. Repeatedly capturing individual frames of ModStealth displaying scenario execution; until acceptable footage was acquired for each scene.
5. Generating video from the collected still frames; conducting post processing, and other video editing as required to achieve final video.

This involved several technical challenges, as described in the following paragraphs.

The DNCW concept involves the rapid creation, employment, and release of “teams” of heterogeneous resources in response to dynamic battlespace events. In the storyboarded scenario, a response is triggered by the creation of a roadblock by insurgents within an urban area (Baghdad), as shown in Figure 3-3. A high value target is seen fleeing the scene in a truck. A team of responders is dynamically assembled as tasked which includes a UAV, a Stryker armored vehicles, and a fighter aircraft, as shown in Figures 3-4 through 3-6. The UAV tracks the truck through the city, and directs the Stryker to intercept it so that the high value target can be captured. Meanwhile, a fighter aircraft is diverted to attack the roadblock.

The first challenge was deciding what method to use for capturing ModStealth video in a useful format. Capturing the monitor's video feed directly was discarded due to the need for specialized hardware/software, the need to run in the Linux environment, and the short turn-around time. After some preliminary investigation, a freeware video capture application, xvidcap, was selected. This software allowed a specific window in the Linux Xwindows environment to be selected. Anything displayed in that window could then be captured as a sequence of JPEG images.

The next challenges were related to how ModStealth itself functions. The first issue was how it displays the Baghdad terrain database: large areas of the terrain would sometimes fail to display, showing blank white expanses instead. This appeared to be due to a memory management problem, as it would become progressively worse the longer the software was used. Scene locations and camera angles had to be carefully selected to avoid these blank areas. In addition, the areas did not remain constant over time. In several cases already-created scenes had to be re-made from scratch because the original location became a blank area between software runs. Another issue was related to ModStealth being CPU intensive. The xvidcap software had to run on the same machine as ModStealth, and the added workload resulted in ModStealth's displayed frame rate dropping considerably during captures. This also caused frequent, and sometimes severe, "warping" behavior by the displayed entities. They would jump from one location to another due to the loss of video frames. There was no good workaround for this behavior, but it could be slightly mitigated through careful selection of camera angles to minimize the amount of background displayed in a given scene. In some cases capturing the same scene multiple times enabled swapping of individual frames to fill in "warped" gaps; at times it became necessary to create filler frames by hand using image editing software. In general creating a subject scenario in good terrain and establishing and capturing enough usable footage for several seconds of display each over numerous scenes was extremely labor-intensive and comprised the majority of the effort.

The camera controls in ModStealth are rather primitive, allowing only fixed views (either stationary or linked to a specified entity). There are no provisions for zooming or panning a given view except in large increments and with large time delays, preventing any sort of smooth application of those filming techniques. Capturing a scene consisted of experimenting until the most favorable camera angle was achieved, and using that angle exclusively throughout a given "take."

Several tools were used in support of this activity. The scenario storyboard was developed using Microsoft PowerPoint. The scenarios were run using the JSAF simulation software in conjunction with the ModStealth 3D visualization software. Still frames of each scene were captured using xvidcap, an open source image capture tool for Linux. Image editing was performed using Corel Paint Shop Pro in the Windows environment and The Gimp (open source) in the Linux environment. Conversion from still frames to video, as well as some post processing and special effects, was accomplished using Adobe Premiere Pro.

In spite of efforts to storyboard the scenario beforehand and get approval, numerous changes in scenario structure, the entities involved, and the scenes portrayed continued to be made throughout the process. After multiple iterations sufficient ModStealth footage was captured to represent all the desired scenes comprising the final form of the scenario. Editing existing images and blending compatible sequences of stills was employed to mitigate "warping" of depicted units. Once the footage was assembled it was handed off to the audio/visual lab for further post processing, additional special effects, and final editing to accommodate numerous additional requested changes.

3.4 TBMCS-SAA Stimulation

In order to create a complete closed-loop simulation, this effort investigated the generation of air track messages by JSAF, and the insertion of such air track messages into the TBMCS Situation Awareness and Assessment (SAA) application's Track Management Data Base (TMDB), so that they could be display as part of the Common Operating Picture (COP). Some of the relevant TBMCS elements and data flows are shown in Figure 3-7. These include:

- **ADSI –** Air Defense Systems Integrator (ADSI), is a real-time tactical command, control, communications, computers, intelligence, surveillance, reconnaissance, and targeting (C4ISRT) system. It operates at both the strategic and tactical levels as a real-time bridge (receives, forwards) between tactical data links and intelligence data sources.
- **SAA –** Situational Awareness and Assessment - SAA is a major component within TBMCS. SAA has two major functions:
    - Situation Awareness, which provides a configurable data display for various users depending on their primary responsibility, and
    - Situation Assessment, which uses the data display and provides analysis tools for experts to support the identification and intent of threats. It provides:

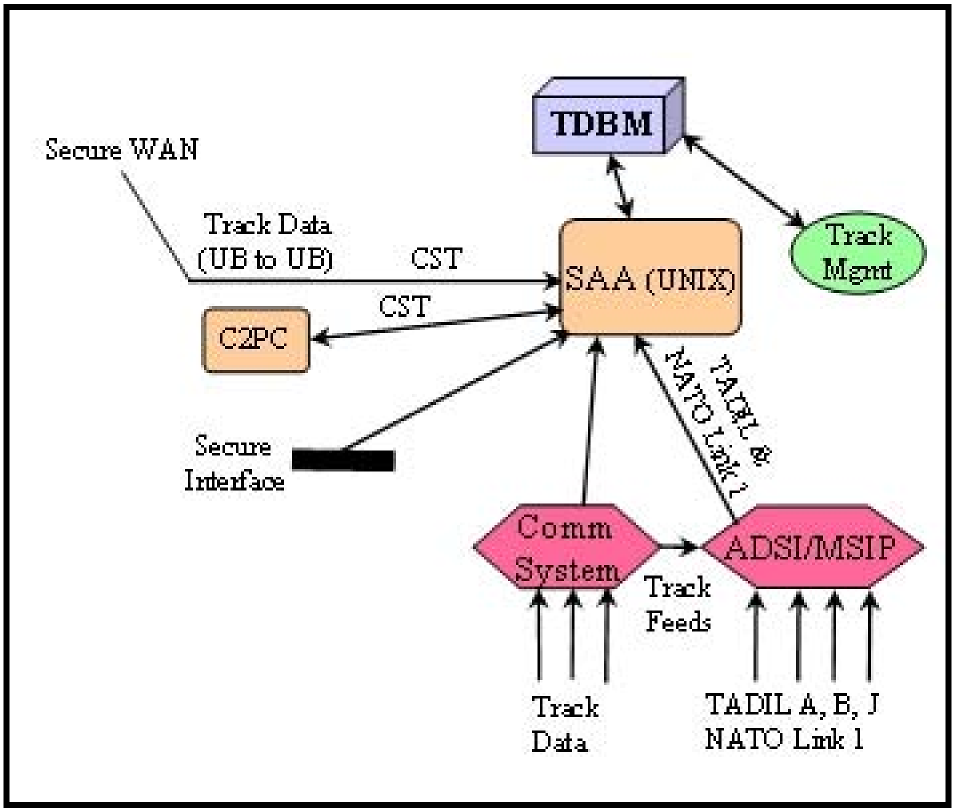


Figure 3-7 Subset of TBMCS Elements and Data Flows

1. Near real-time views of theater air and ground tracks for the theater war-fighting commander (e.g., the JFC, the JFACC and their staffs).
2. Textual and graphical information for Friendly Order of Battle (FrOB) and near-real-time Intelligence data.
3. A graphical display – all-source-correlated information about enemy forces, including electronic combat information, to assist in analysis and evaluation of enemy threat status.

SAA receives, processes, and correlates multiple source/sensor inputs. The correlation process provides the capability to generate new tracks, update existing tracks, label tracks as ambiguous, re-correlate previous ambiguous tracks, manually modify existing tracks and manually merge tracks.

- **TDBM/TMDB** – (Track Database Manager/Track Management Database). A flat file database that stores tracks. The interface between SAA and TDBM is internal to TBMCS.
- **Track Management Service –** The Track Management Service provides an API for the retrieval, addition, and modification of Track data stored in the Track Data Base Manager (TDBM). Track addition and modification supports Unit, Platform, and Emitter tracks.

Two approaches to passing air tracks from JSAF to TBMCS were investigated: 1) generating TADIL-J air track messages from the entity state messages output by JSAF; and 2) using the TBMCS 1.1.3 Track Management Service to create air tracks, using entity state data generated by JSAF via the HLA-to-XML Gateway. Each of these approaches is discussed below.

3.4.1 TADIL-J Message Generation

This activity involved developing a process whereby entities simulated in JSAF are represented on the Common Operating Picture (COP) display as though they were tracks of actual aircraft/vehicles. This consisted of the following steps:

1. The primary focus of this task was to create tracking data of the same form as that generated by tracks of real-world entities. This data could then be fed into the COP system in the same manner as real-world data, obscuring the actual source and enabling seamless blending of simulated and real-world tracks.
2. Since JSAF does not generate such tracking data, an intermediate application (or series of applications) was needed to accept the entity data that JSAF does generate, and convert that data into properly-formatted tracks. A software application called SIMPLE, developed by the National Simulation Center and available as GOTS, was used for this purpose.

This involved several technical challenges, as discussed in the following paragraphs.

SIMPLE came with minimal documentation in the form of a Program of Instruction (POI) which was very linear: it walked the user through a series of steps for setting up a single type of installation to address a single type of simulation environment. This POI was oriented towards simulating ground units in JCATS. Interfacing with JSAF, to generate air tracks, was not addressed. In addition, the POI was outdated; some of the system configuration information was incorrect for the latest version of SIMPLE. Some relief was available via e-mail to the points of contact listed on the web site, but in general the software had to be set up via trial and error.

In general, SIMPLE functions by receiving entity state and event information from the simulation software being used (such as JSAF), comparing that information against a local MySQL database of unit types and structures, and then generating appropriate messages formatted as though they were being generated by the corresponding real-world entities. The local database can be automatically populated via scenario data from a simulation such as JCATS, but only for a select set of ground units of interest (most commonly artillery, judging from the content of the POI). In the case of interfacing with JSAF, specifically with air units, the population had to be accomplished by hand. For example, populating the database with a single Soviet MiG-23 Flogger aircraft required the following steps:

1. The entity first has to be entered into the “AggEdit” screen, as shown in Figure 3-8. The software does not appear to be sensitive to the values in the Name fields, although this has not been exhaustively tested. Some of the other fields are self-explanatory, while others were populated by “best guess.”

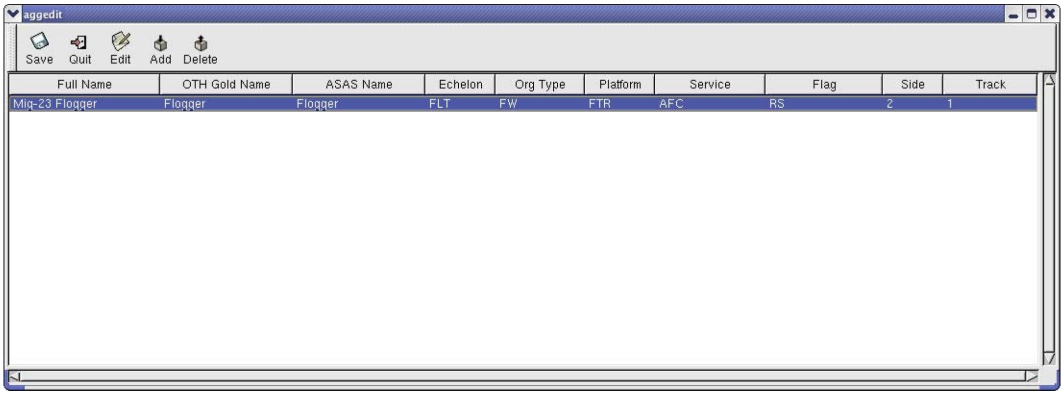


Figure 3-8. SIMPLE AggEdit Screen

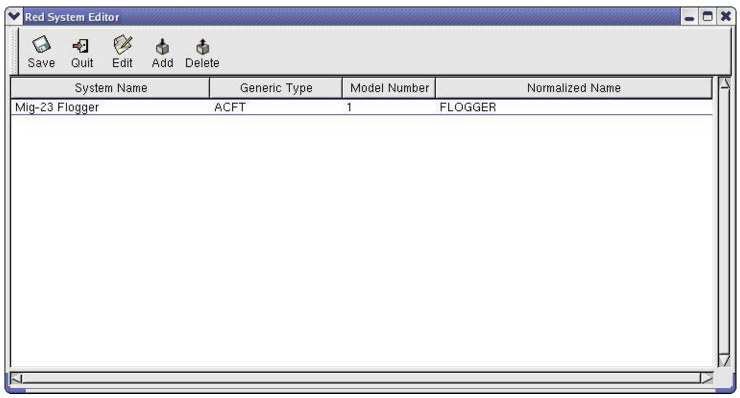


Figure 3-9. SIMPLE RedSys Screen

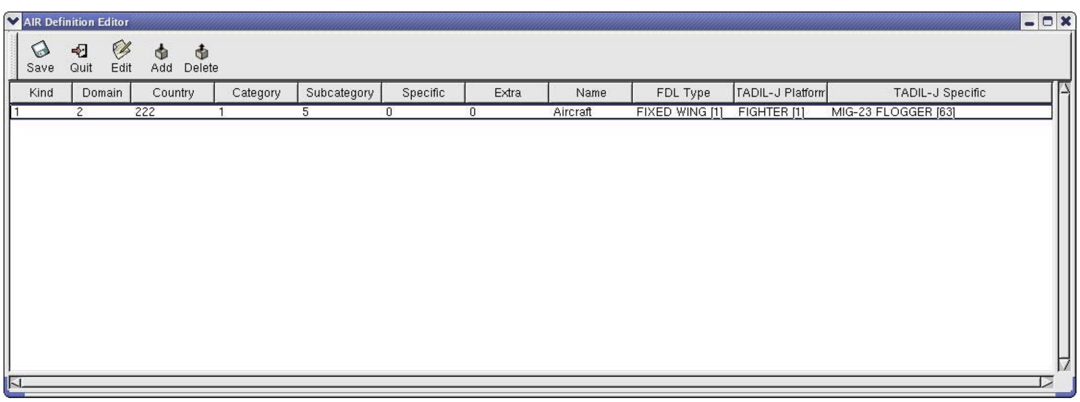


Figure 3-10. SIMPLE AirDef Screen

2. The "RedSys" screen, shown in Figure 3-9, identifies the given entity type as "red force" or enemy. Again, the Name field does not appear to have any effect on the simulation results. The Normalized Name is populated via a pick list and appears to be the primary method used by SIMPLE to identify which entity applies to this entry.

3. Finally the “AirDef” screen, shown in Figure 3-10, allows detailed definition of the specific aircraft entity type. The values are taken from the standard DIS Enumerations of the given unit type, which can be obtained either through the DIS documentation or by a command-line query from inside JSAF itself. The latter method was preferred so as to insure compatibility with JSAF’s chosen enumerations.

Initial experimentation indicated that SIMPLE would generate TADIL-J tracks for entities which do not have a corresponding entry in the local database, though some detailed information may then be lacking. More detailed testing needs to be done once a reliable method for decoding the content of the TADIL-J track messages is obtained.

The primary method for SIMPLE to communicate with JSAF is for SIMPLE to run as a federate in an HLA federation including JSAF. However, this was not possible in this case, because SIMPLE was hard-coded to require the RTI-NG, rather than the RTI-S version currently used within the JSB-RD environment; the two are incompatible. Fortunately SIMPLE will also accept DIS PDUs, if configured as shown in Figure 3-11. Of particular note is the port and address configuration at the top of the window: this must be set to the broadcast address of the local network, not the specific IP address which is generating PDUs. The various fields referring to Federation information are not used and can be left blank if desired.

To support this configuration, the HLA-to-DIS Gateway software must also be properly configured. Through experimentation it was determined that a script file to successfully execute the gateway would look like the following:

```
#! /bin/sh

cd /opt/jsaf2004.wp/build/JSAF/src/Gateway

./gateway -terrain iraq_v4d+_050808 -disport 3333 -forced_ddm_subscriptions
-rid_name ../../../../../../home/dysonm/jsaf/RTI-s_1.3_D10A.rid
```

The `–rid_name` argument must specify the full path name to the RID file, which was locally stored. Also note that the `–disport` argument has to match the port number configured in SIMPLE for accepting DIS broadcasts.

SIMPLE generates a myriad of different message types. The goal of this activity was to generate TADIL-J air track messages. The SIMPLE screen for specifying the TADIL-J reporting parameters is shown in Figure 3-12.

The fields are largely self-explanatory. In this case, the “sensor” is defined as a static point location at a specific lat/long/altitude, with a max range of “0.0” which is treated as infinite. It’s a simple matter to define the sensor as an entity within JSAF, instead, by matching the entity’s DIS values and markings within the configuration screen. The resulting TADIL-J message types include J2.5, J3.2 and J3.5 messages, which are sent to the specified IP address and port. Up to eight sensors of this type can be defined in SIMPLE, and each can transmit to a different destination if desired. The content of the TADIL-J track messages generated by SIMPLE were verified using Northrop Grumman's proprietary Gateway Manager software.

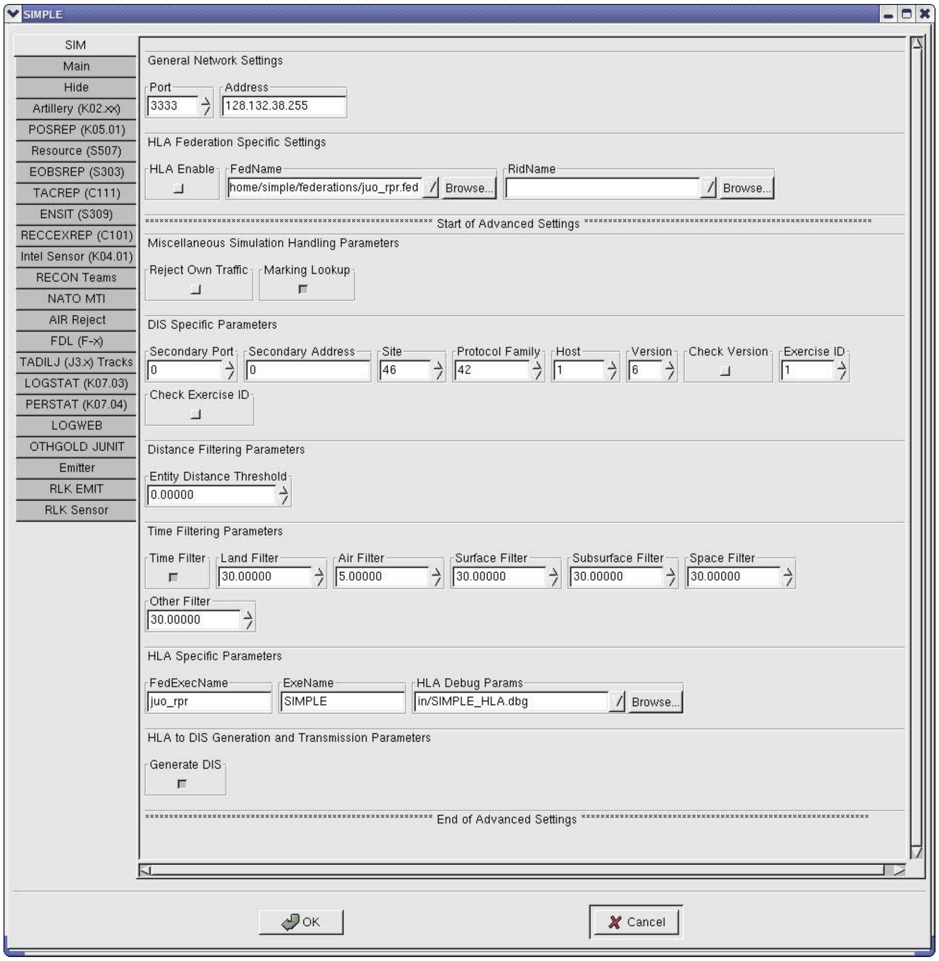


Figure 3-11. SIMPLE DIS-Compatible Configuration

The JSAF->HLA-to-DIS Gateway->SIMPLE configuration has successfully undergone preliminary tests containing differing numbers and types of aircraft. However, no local capability yet exists to decode SIMPLE's TADIL-J output, which uses the GCCS/MTC variant of the Socket-J format. The Tactical Information Processor & Online Fusion Facility NT (TIPOFFNT) application was obtained and installed in the classified lab. This fielded application interfaces with communication systems. However, it was unable to successfully process the TADL-J messages generated by SIMPLE. The results were ambiguous: TIPOFFNT accepted the output from SIMPLE, but yet did not produce any positive results. TIPOFFNT did, in fact, reject other input formats, so it apparently does perform error checking. Finally, SIMPLE generated test data was sent to the developers of the Northrop Grumman Gateway Manager software, who then returned the successfully decoded track data. A local capability for decoding air track data in GCCS/MTC format is actively being sought.

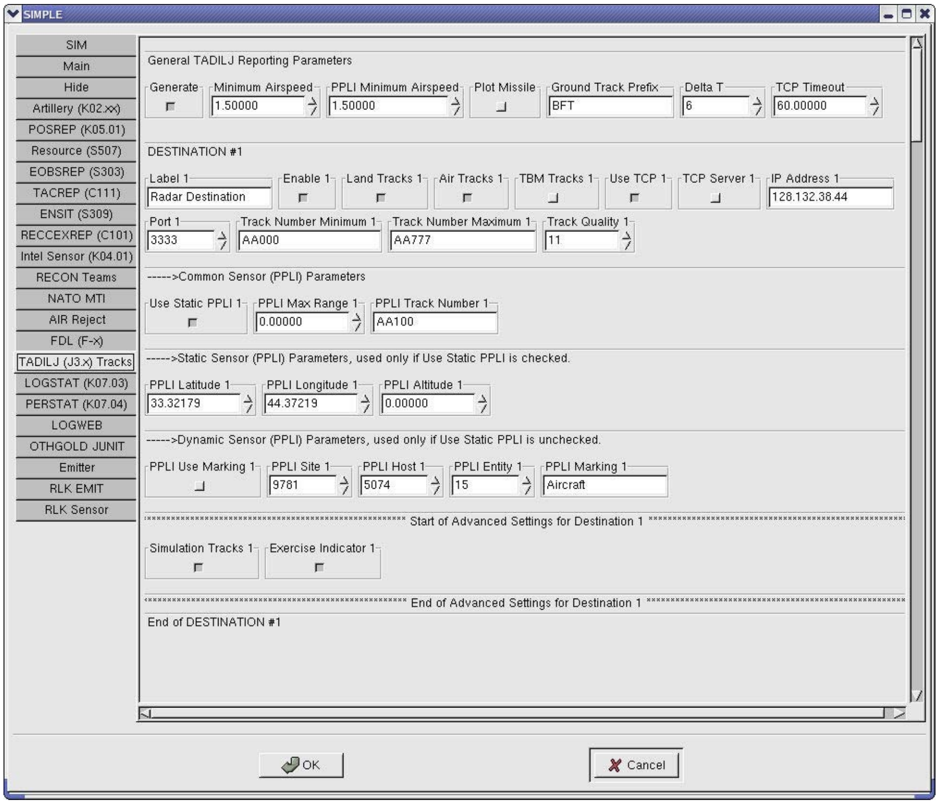


Figure 3-12. SIMPLE TADIL-J Reporting Parameter Screen

The existing configuration can successfully generate track data for JSAF aircraft which is indistinguishable from real-world tracks. For the tracks to be ingested and then displayed on the COP, either a functioning ADSI, an instance of GCCS, or some alternative that fulfills a similar function is required. This requirement must be met for the COP to display tracks from any external source, including from real-world data.

3.4.2 Using TBMCS 1.1.3 Track Management Services

This activity involved investigating the use of the TBMCS 1.1.3 Track Management Service to add, modify, retrieve, and remove tracks from the TMDB. This consisted of modifying the HLA-to-XML Gateway to invoke the TBMCS 1.1.3 Track Management Service. PAR/C3I and Lockheed Martin personnel, as well as Northrop Grumman personnel in Colorado Springs, provided invaluable assistance in this activity.

Using the JMTK-based SAA COP graphical user interface to manually enter track data, it was found to be possible to successfully add, update, and delete tracks.

Using the Track Management Service APIs, it was found to be possible to successfully add and delete tracks in the TBMCS Track Management Data Base (TMDB). It was also found to be possible to modify a track's name/call sign and metadata. However, it was not possible to update the location of an existing track. Figure 3-13 shows a side-by-side comparison (with slightly different map scales) of the COP display (on the left) with the JSAF Plan View Display (on the right), both showing two aircraft (one red and one blue).

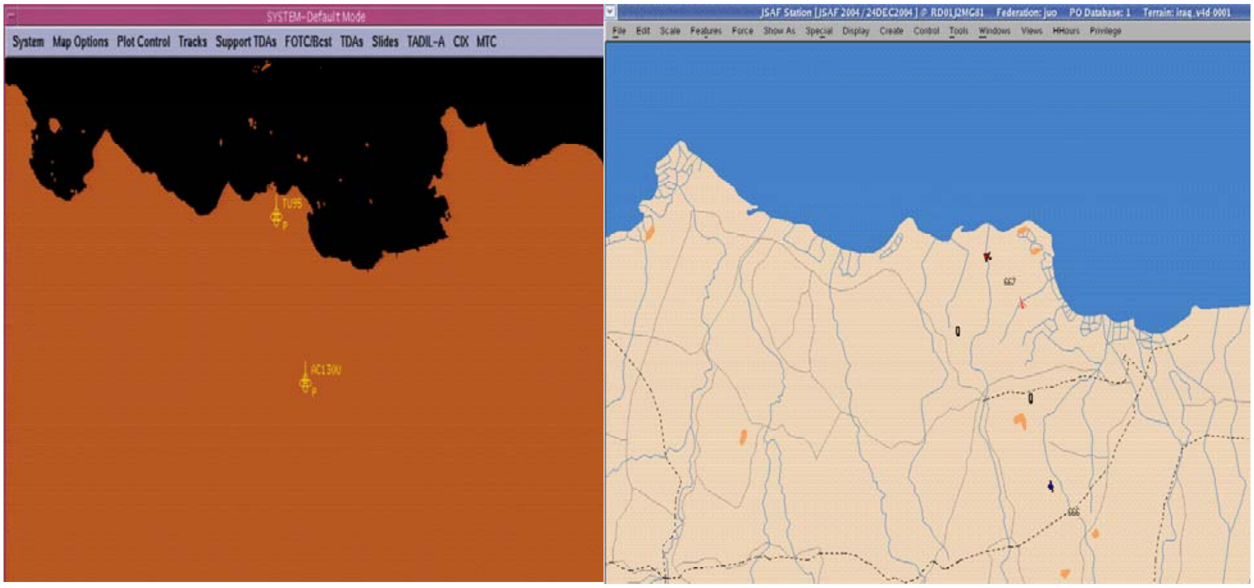


Figure 3-13: COP and JSAF Map Displays

Using the TBMCS 1.1.3 Track Management Services was found to have several significant limitations:

1. As noted above, the Track Management Service cannot update a track's position. The track must be deleted and then recreated with a new location. This prevents the retrieval of a track's history from the TMDB. No way to retrieve history of tracks even when edited thru the JMTK client on the TBMCS Universal Build.
2. Not all methods of the services have been implemented; and some methods were unable to support the rapid input of tracks via this mechanism. After feedback was provided to Lockheed Martin on the problems that were encountered, Lockheed Martin stated, "The TBMCS Track Service is not designed to handle high performance loads. This is the reason that Link-16 tracks are not supported."
3. The Web Service was unable to perform any operation other than retrieving tracks. This appears to be due to a misconfigured deployment descriptor in the WebLogic configuration. To obtain the full functionality of the Track Management Service the Web Service interface was bypassed in favor of the underlying Enterprise Java Beans (EJB) interface.
4. Sporadic failure of the TDBM database. The TDBM database becomes inaccessible to both the Web-Service and EJB after a period of time. The returned error message indicates a timeout problem. Restarting the SAA server seems to be only workaround. The cause of these failures remains unknown, and the Track Management Service cannot be ruled out as the source of the problem.
5. The existing Track Management Service API is incomplete – only three track types are supported: emitter, platform and unit tracks; the existing methods are simplistic; track locations cannot be updated, etc.

6. If certain attribute fields in a track object are not filled in, the track cannot be retrieved using the getTracks method.
7. The Track Management Service does not allow the user to assign an id to a track; the id is assigned by the server. In order to delete a track, you must first retrieve the track to get its id.
8. The performance of the Track Management Service is very poor. Adding, updating, or deleting a track has an elapsed time of more than one second. Even with only a small number of entities, the Track Management Service was unable to keep up with JSAF entity updates. Several attempts to improve performance (using a faster processor, adding physical memory, increasing process priorities) had no appreciable effect. Lockheed Martin stated that the Track Management Service never had any performance requirements.

A potential alternative, which has yet to be investigated, is to use the GCCS-M Track Management Service API to interact with the TBMCS TDBM. An initial review of the available documentation indicates that this API is much more complete and robust.

4. LESSONS LEARNED

This section summarizes the lessons learned during this effort. The key areas of lessons learned under this effort include:

- JSAF and associated simulations,
- Scenario preparation,
- HLA-to-XML Gateway software,
- Integrated Situation Viewer software,
- DNCW visualization,
- TBMCS SAA interface.

Each of these discussed in the subsections below. Finally, conclusions and recommendations are given.

4.1 JSAF and Associated Simulations

JSAF is a very large and complex piece of software. JFCOM J9 maintains a staff of more than 30 developers who are constantly modifying and extending JSAF to meet the requirements of an ongoing series of simulation experiments. Although the JSAF development staff has been helpful in finding answers to questions, supporting external users of JSAF is not a high priority. There is no JSAF help desk, or any other similar support mechanisms.

The JSAF software contains an impressive amount of functionality. However, much of the functionality that appears to be present is difficult or impossible to access in practice. JSAF has been modified extensively over the years to meet specific needs of particular exercises and experiments. Many of these modifications were never successfully completed. However, they remain in the JSAF software distribution. In some cases, it appears that capabilities were added at one point, but were later incompletely removed, leaving unused entries in data files, etc. For example, JSAF contains a number of software libraries that appear to support Link-16/TADIL-J communications functionality, as well as data elements. The MC02 FOM includes messages that describe Link-16 radios and TADIL-J messages. However, all attempts to make use of this functionality have been unsuccessful.

JSAF does a reasonable job of simulating the movement of physical entities from one location to another. In particular, the version used under this effort (JSAF 2004) has an improved aircraft flight dynamics model. It also does a reasonable job of simulating many types of individual weapons and the related combat activities. It can model tactical aircraft and simple air missions adequately. Although it does model some types of sensors, including the detection of targets, it does not model the tracking, reporting, or dissemination of information resulting from sensor detections to other entities. It does not model most ISR assets (e.g., AWACS, JSTARS, UAVs) with useful realism.

JSAF has two distinct interfaces for creating scenarios. One is the interactive Unit Editor, which allows military units and platforms to be created, placed on the Plan View Display, and assigned tasks. Once created, scenarios can be saved to files for reloading later. However, in working on the SAB demonstration and the DNCW visualization, this mechanism was found to be problematic. Entities moving along a defined path would often not be restored to their correct state after a save. Often entities would be reset to the beginning of the path, or would be unable to complete assigned tasks.

The second interface is the Spreadsheet mechanism, which allows spreadsheets specifying air mission information to be read in, from which the necessary aircraft are created and then tasked. Spreadsheets can also be written out, but without any tasking information. This is the mechanism that has been used to create air mission scenarios created from Air Battle Plans extracted from TBMCS. However, the JSAF spreadsheet mechanism has also proved to be problematic. Specifying a time on target for an air mission does not work correctly. The FWA Hold task frame, which is used to take up slack time between other tasks, also does not function correctly; instead of orbiting a point, the aircraft simply fly in a straight line on their current heading. Although numerous experiments have been performed, no mechanism for accurately controlling the timing of air missions has yet been discovered.

Oddly, there is very little overlap between the aircraft tasks that can be assigned using these two mechanisms. Tasks that can be assigned interactively using the Unit Editor generally cannot be assigned using the Spreadsheet mechanism, while the mission types that can be assigned using the Spreadsheet mechanism cannot be assigned using the Unit Editor. The Unit Editor also offers a number of interactive commands (e.g., Fly Higher/Fly Lower) that cannot be accessed through other mechanisms. In particular, there has been no mechanism that allows military units to be tasked via an HLA message from another federate. JFCOM controls JSAF-based scenarios using multiple operators, who interpret commands and interactively manipulate the JSAF entities accordingly. Because of this, JFCOM sees little need for such a capability.

Under a related effort, SAIC has developed a modification to JSAF which allows entities to be retasked using text messages from other applications. This mechanism is currently used by the Integrated Situation Viewer to perform limited retasking of aircraft. This tasking mechanism bypasses both the JSAF Unit Editor and the spreadsheet mechanism to directly manipulate the task frame stack of the entity. This has the potential to provide more precise control of JSAF aircraft, as well as other entities.

4.2 Scenario Preparation

The existing TBMCS-to-JSAF capability extracts an Air Battle Plan from the TBMCS Air Operations Data Base (AODB), along with supporting target information from the MIDB, and generates a JSAF input spreadsheet that can be used to execute those air missions. The Route Planner tool is used to add intermediate waypoints to the route for each air mission. This is a very powerful mechanism.

There remain several limitations to this capability that should be addressed. Currently, multiple queries are performed directly on AODB and MIDB tables. TBMCS 1.1.3 provides a collection of web services which can be used to access some of this information; however, these services are not yet complete. Also, TBMCS 1.1.4, which includes TBONE, is beginning to come online, with a completely different database structure and a different set of access services. Due to the database differences, the existing Air Battle Plans and other information in the TBMCS 1.1.3

databases will almost certainly not be migrated to TBMCS 1.1.4. At some point in the near future, this information will be lost if it is not saved in some other form.

There are also several issues concerning the translation of ABP information into JSAF input. AODB aircraft types must be translated into corresponding JSAF entity types. The necessary JSAF entities are created for each specified air mission. This makes it difficult for a particular aircraft to fly multiple missions in sequence. ABP mission numbers are not used by JSAF. Instead, the call signs for each mission in the ABP are augmented with unique numeric IDs, and are inserted into the "markings" field of the JSAF entities. These call signs are not always unique. The mission types specified in the ABP must be translated into JSAF air mission task frames. Currently, the task frames used are Interdiction/Attack Ground for strike missions, FWA Hold for orbiting (including CAP, tankers, etc.), and Return to Base. Finally, the route points that can be entered into JSAF are two-dimensional (latitude-longitude). Altitude is specified indirectly, as a parameter of the top-level task frame. This makes it difficult to specify mission profiles that depend on variations in altitude.

4.3 HLA-to-XML Gateway

The HLA-to-XML gateway allows multiple applications, such as the Viewer, to access the stream of information coming out of the simulations without having to deal with the complexities of HLA. However, it currently operates in only one direction. Therefore, it only supports "receive-only" clients. Any HLA messages that an application needs to send back to the simulation to, for example, retask an aircraft, must use another mechanism.

4.4 Integrated Situation Viewer

Improvements to the Integrated Situation Viewer under this effort were limited. Some improvements in performance were derived from the use of updated versions of the JView visualization software. Map background loading was also significantly improved, due to the availability of a local repository of NGA CADRG and DTED data. A Java Naming and Directory Interface (JNDI) capability that was expected to become available in JView, so that the Viewer could support multiple displays easily was never completely finished.

4.5 DNCW Visualization

During the development of the DNCW visualization, it was learned that it is very import to "audition" JSAF entities and their behaviors as early as possible during the planning of such a scenario. As soon as a particular entity type is identified as a candidate participant in a scenario, its support within JSAF should be checked, so that alternatives can be investigated if necessary. The required behaviors of each entity type should also be thoroughly tested, as the implementation of behaviors in JSAF can often be problematic. It may not be possible to perform the behavior as originally envisioned, and workarounds or alternatives may have to be developed. In particular, combat behaviors need to be checked, as they can be very sensitive to the timing of the detection of the target entity by the firing entity.

Similarly, it is also necessary to "audition" the key locations where the scenario events are to take place. Problems with the connectivity of the road network database, which are invisible in both the JSAF Plan View Display and the ModStealth 3D display, may disrupt the specification of movement paths, resulting in entities either refusing to move, or moving in an unrealistic manner. Also, the ModStealth application appeared to have memory management problems when used with the very large, dense Baghdad database. Sections of the database would progressively disappear from the 3D display.

Finally, a key lesson learned in the production of the DNCW visualization is that it is critical to obtain signoff on an initial storyboard form of the scenario visualization before doing extensive modeling and video capture efforts. Changes to the scenario, or to the visualization of the scenario, after production has begun are extremely problematic, resulting in extensive rework.

4.6 TBMCS SAA Interface

The effort to create Link-16 air track messages from JSAF-generated aircraft entity state information and to feed those messages into the TBMCS SAA application proved to be far more difficult than expected. Although it is a fielded system, the various interfaces to TBMCS 1.1.3 proved to be very difficult to work with. Documentation was limited or nonexistent. The available web services turned out to be incomplete, immature, and very slow. However, it was eventually possible to completely close the loop between JSAF and TBMCS, at least in a limited fashion that can be expanded in the future.

4.7 Conclusions and Recommendations

The conclusions and recommendations resulting from this effort are summarized below:

1. JSAF remains poorly suited for use within a small, dynamic laboratory environment. It requires a large, highly trained, and highly skilled staff to maintain, to operate, and especially to modify or extend it. Extending or modifying the JSAF software in support of a specific simulation experiment is difficult and time-consuming. Although it can be integrated with other existing tools and C4ISR systems, this is not easy to accomplish. Continued use of JSAF within the AFRL/IFSB modeling and simulation laboratory will require a significant commitment to develop and maintain a skilled and experienced staff.
    a) JSAF, and its associated applications, should be kept up to date on a regular (semi-annual or annual) basis. This is necessary in order to remain reasonably in synchronization with the JSAF development staff at JFCOM.
    b) The JSAF software should continue to be investigated, and to be modified and extended as needed. This includes:
        i) Fixing the spreadsheet input mechanism to better support the execution of air missions.
        ii) Creating new air mission task frames, and modifying existing air mission task frames.
        iii) Extending the existing retasking mechanism to also support initial entity creation and tasking, replacing the spreadsheet mechanism.
    c) Enhance the JSAF-GIESim communication capability allow JSAF aircraft to send and receive various types of Link-16 messages in response to simulation events (e.g., takeoff, attack, landing) or time intervals (e.g., PPLIs).

d) Other more modern entity level simulations that support air operations should continue to be sought to augment or replace JSAF. In particular, the Army's OneSAF Objective Objective System (OOS) should be investigated as a potential platform for modeling Air Force platforms and systems. However, OOS now appears to be nearly two years behind its original development schedule. An initial operating capability still has yet to be released.

2. The Scenario Preparation software should be refactored to separate the importing of information from TBMCS 1.1.3 and the generation of JSAF-specific input. This will allow scenario information to be imported from other sources, as well as the generation of input data for other simulations. Data import tools and simulation input generation tools should be built around a common scenario preparation database that contains friendly force data, adversary force data, and scenario specific plans and tasking.

   a) The Air Battle Plans and associated supporting data contained in the TBMCS 1.1.3 AODB and MIDB should be extracted and stored in the above scenario preparation database while TBMCS 1.1.3 is still available within the laboratory, as this data is unlikely to be migrated to TBMCS 1.1.4.

   b) Air Control Measure information for the Air Battle Plans should also be extracted from the AODB. This information should be used by the Router to create more realistic routes, and should be displayable by the Viewer.

   c) The Route Planner should be updated to make the routes that it produces more realistic.

   d) Enhance the Scenario Preparation software to automate the generation of all necessary GIESim scenario input files, including network design and entity identifier mapping. This will facilitate the development of integrated JSAF-GIESim scenarios.

3. The HLA-to-XML Gateway should be re-examined to determine if this is still the best approach to supporting HLA clients within the JSB-RD facility, rather than creating multiple HLA federates as needed.

   a) The HLA-to-XML Gateway should publish simulation events that correspond to all planned mission events, including takeoff and landing, weapon fire and munition detonation, start and end of refueling, etc., so that they may be accessed by client applications.

4. The Integrated Situation Viewer should be enhanced to display additional air mission-related information, and to display information in additional forms. This includes:

   a) Adding support for Java Naming and Directory Interface (JNDI) technology, to allow all types of views to be duplicated across multiple monitors on multiple systems, while remaining synchronized with one another.

   b) Expanding support for retasking air missions.

   c) Displaying entity interactions, including weapon fire and munition detonation, and takeoff and landing, as well as entity removal.

d) Displaying planned mission information from the Air Battle Plan, and expanded graphical comparison of planned mission states with actual mission states.

e) Displaying time line/Gantt chart type displays showing planned and actual mission event (takeoff, on target, return, etc.) times.

f) Additional graph and chart views which show aggregate mission success, in terms of aircraft lost and/or targets destroyed or disrupted.

g) Display of Air Control Measures, which show restricted airspaces.

h) Display of weather conditions, in 2D or 3D, as published by OASES.

i) Three-dimensional views that can be attached to a specific aircraft or group of aircraft.

j) Improved controls for customizing individual views, as well as creating, naming, storing, and retrieving collections of views including view configuration (i.e., window size and placement) information.

5. ACRONYMS

This section provides a listing of acronyms and their meanings as used in this document.

AAA	Anti-Aircraft Artillery
ABP	Air Battle Plan
ACE	Air and Space Constructive Environment
ADSI	Air Defense Systems Integrator
AFAMS	Air Force Agency for Modeling and Simulation
AFRL	Air Force Research Laboratory
AF/XPXC	Air Force Strategic Planning Directorate, Future Concept Development Division
ANT	Airborne Networking Technology
AODB	Air Operations Data Base
API	Application Programmer Interface
AWACS	Airborne Warning and Control System
AWSIM	Air Warfare Simulation
C2PC	Command and Control for the PC
C3	Command, Control, and Communications
C4ISR	Command, Control, Communications, Computers, Intelligence, Surveillance, and Reconnaissance
C4ISRT	Command, Control, Communications, Computers, Intelligence, Surveillance, Reconnaissance, and Targeting
CADRG	Compressed ARC Digital Raster Graphics
CAP	Combat Air Patrol
CDRL	Contract Data Requirements List
CGF	Computer Generated Forces
CLIN	Contract Line
COA	Course of Action
COP	Common Operating Picture
CPE	Commander’s Predictive Environment
CTDB	Compact Terrain Data Base
DARPA	Defense Advanced Research Projects Agency
DBST	Digital Battlestaff Sustainment Training
DIS	Distributed Interactive Simulation
DMSO	Defense Modeling and Simulation Office
DMT	Distributed Mission Training
DNCW	Dynamic Network Centric Warfare
DoD	Department of Defense

DSAP	Dynamic Situation Awareness and Prediction
DTED	Digital Terrain Elevation Data
DTSim	Dynamic Terrain Simulation
EBO	Effects Based Operations
EBV	Entity Bit Vector
EJB	Enterprise Java Beans
EOB	Enemy Order of Battle
FOM	Federation Object Model
FrOB	Friendly Order of Battle
FTP	File Transfer Protocol
FWA	Fixed Wing Aircraft
GCCS	Global Command and Control System
GCCS-M	Global Command and Control System – Maritime
GIE	Global Information Enterprise
GIESim	Global Information Enterprise Simulation
GOTS	Government Off The Shelf
GUI	Graphical User Interface
HE	Human Effectiveness Directorate (of AFRL)
HLA	High Level Architecture
IFSB	C4ISR Modeling & Simulation Branch
IF	Information Directorate (of AFRL)
ISR	Intelligence, Surveillance, and Reconnaissance
JBI	Joint Battlespace Infosphere
JCATS	Joint Conflict and Tactical Simulation
JDBC	Java Database Connectivity
JFACC	Joint Force Air Component Commander
JFC	Joint Force Commander
JFCOM	Joint Forces Command
JMTK	Joint Mapping Tool Kit
JNDI	Java Naming and Directory Interface
JPEG	Joint Photographic Experts Group
JSAF	Joint Semi-Automated Forces
JSB-RD	Joint Synthetic Battlespace for Research and Development
JSIMS	Joint Simulation System
JSTARS	Joint Surveillance and Target Attack Radar System
JUO	Joint Urban Operations
LOS	Line of Sight
MARCI	Multi-system Automated Remote Control and Instrumentation
MC02	Millenium Challenge 2002

METOC	Meteorological/Oceanographic
MIDB	Modern Integrated Data Base
MiG	Mikoyan-Gurevich
M&S	Modeling and Simulation
ModSAF	Modular Semi Automated Forces
MTC	Multi-TADIL Capability
NATO	North Atlantic Treaty Organization
NGA	National Geospatial Intelligence Agency
NGMS	Northrop Grumman Mission Systems
NSC	National Simulation Center
OASES	Ocean, Atmosphere, and Space Environmental Services
PBA	Predictive Battlespace Awareness
PDU	Protocol Data Unit
PFM	Pressure Field Modification
POI	Program of Instruction
PSM	Portable Space Model
PVD	Plan View Display
RID	RTI Initialization Data
RPR	Realtime Platform Reference
RTI	Run Time Infrastructure
SAA	Situation Awareness and Analysis
SAB	Science Advisory Board
SAM	Surface to Air Missile
SIMPLE	Simulation to C4I Interchange Module for Plans Logistics and Exercises
SNE	Synthetic Natural Environment
SNN	Simulation Network News
STOW	Synthetic Theater of War
TAOS	Total Atmosphere Ocean Services
TADIL-J	Tactical Data Link - Joint
TBMCS	Theater Battle Management Core System
TBONE	Theater Battle Operations Net-Centric Environment
TDBM	Track Database Manager
TIPOFFNT	Tactical Information Processor & Online Fusion Facility NT
TMDB	Track Management Database
UAV	Unmanned Aerial Vehicle
UB	Universal Build
UCAV	Unmanned Combat Aerial Vehicle
USSPACECOM	U.S. Space Command
VIP	Very Important Person

WAN	Wide Area Network
XML	eXtensible Markup Language